NAVAL HEALTH RESEARCH CENTER

TECHNICAL MANUAL FOR THE NAVY COMPUTER ASSISTED MEDICAL DIAGNOSIS KNOWLEDGE BASE EDITOR (NCAMD-KBE)

Version 1.0.

H. L. Ly

D. M. Pearsall

19961004 067

Technical Document 96-4D

NAVAL HEALTH RESEARCH CENTER
P. O. BOX 85122
SAN DIEGO, CALIFORNIA 92186 – 5122

NAVAL MEDICAL RESEARCH AND DEVELOPMENT COMMAND
BETHESDA, MARYLAND

Technical Manual for the Navy Computer Assisted Medical Diagnosis Knowledge Base Editor (NCAMD-KBE), Version 1.0.

Prepared by:

Hoa L. Ly
Dianna M. Pearsall

Naval Health Research Center
Medical Information Systems and
Operations Research Department
P.O. Box 85122
San Diego, CA 92186-5122

Technical Document 96-4D supported by the Naval Medical Research and Development Command, Bethesda, MD, Department of the Navy, under Work Unit No. 63706N M0095.005-6103. The views expressed in this article are those of the authors and do not reflect the official policy or position of the Department of the Navy, Department of Defense, nor the U.S. Government.

SUMMARY

This technical manual contains the information on the program source code, data elements, and the database structure needed to maintain the Navy Computer Assisted Medical Diagnosis Knowledge Base Editor (NCAMD-KBE). This documentation was created using the FOXDOC Version 2.5a program.

TABLE OF CONTENTS

Introduction

This document is the technical guide for the Knowledge Base Editor within the Navy Computer Assisted Medical Diagnosis (NCAMD) System. FoxDoc Version 2.5a was used to generate the program code in Section VII. This technical manual describes the knowledge base data structures and reviews the diagnostic algorithm and data flows. It contains seven sections:

1. System Summary
2. Menu Summary
3. Screen Summary
4. Data Dictionary
5. Tree Diagram
6. Procedure Summary
7. Source Code Program Listing

Section I. System Summary. See the tree diagram in Section V for programs, procedures, functions, and file formats.

```
This system has:

  13520 lines of code
      1 program file
     32 procedure files
    178 procedures and functions
     16 table/dbfs
     17 index files
      1 menu file
     23 screen files
      2 other files
    612 cross-referenced tokens
```

Section II. Menu File Summary. The system has one menu: KBMENU.MNX.

MENU OPTIONS	**MENU OPTION IDENTIFIERS**
System	
Help F1	System_MST_HELP

Backup	
Restore	
Knowledge Base	KNOWLEDGEB
Knowledge Base Area	
Question - Data	
Dictionary	
Exit system	

Section III. Screen Summary. The system has twenty three (23) screen files: BACKUP, KB, KBDEL, QUAL, GOAL, DISPLAY, DISEASE, ACTION, ACTELSE, CLAUSE, RULE, TERM, OBJECT, ENUM, RESTORE, DD, PLAN, DICT, KBEDIT, OBLIST, DDEDIT, DDENUM, and KBLOAD.

A. BACKUP.SCX **Last updated: 10/03/94 at 15:01**

```
 0  2: message...........................
 1
 2
 3   5: mfile...............    To drive:.  1: mdrive.......
 4   .......................
 5   .......................
 6   .......................    Directory:  3: mpath........
 7   .......................
 8   .......................
 9   .......................                    «Backup»
10   .......................
11   .......................                    <Cancel>
12
13
14  Backup file name:.6: mfname............................
```

Window name: W_backup
Coordinates: FROM 0,0 TO 0,60
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	mdrive	Popup	"@^ "
2:	message	Field	
3:	mpath	Popup	"@^ "
4:	maction	Push button	"@*HT \!\<Backup;\<Cancel"
5:	mfile	List	"@&N"
6:	mfname	Field	
7:	mdrive	Popup	"@^ "
8:	message	Field	
9:	mpath	Popup	"@^ "
10:	maction	Push button	"@*VT \!\<Backup;\<Cancel"
11:	mfile	List	"@&N"
12:	mfname	Field	

B. KBDEL.SCX **Last updated: 10/03/94 at 15:01**

```
                    Knowledge Base Delete
0
1  Delete:  2: name...................
2
3
4          <  OK  >  <Cancel>
```

	Name	Type	Picture
1:	mbutton	Push button	"@*HT OK;Cancel"
2:	area.name	Field	
3:	mbutton	Push button	"@*HT OK;Cancel"
4:	area.name	Field	

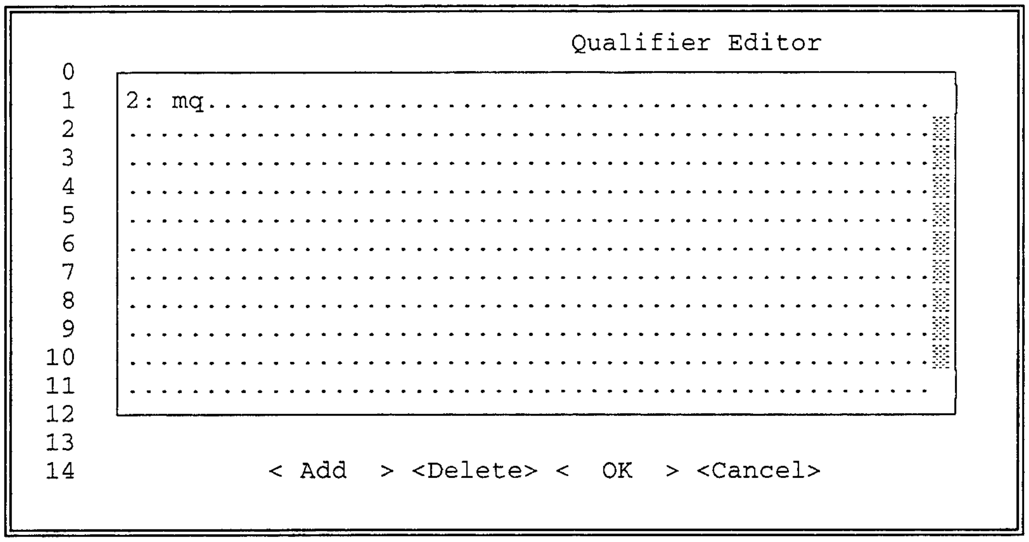
C. KB.SCX   Last updated: 10/03/94 at 15:01

```
 0
 1 <Knowledge Base> 5: name..........................
 2                                                              <   Next    >
 3 ┌──Display thresholds──┐    Rules     6: r
 4 │                      │                                     < Previous  >
 5 │ Consider  2: thr     │    [ ] DIAGNOSIS
 6 │                      │                                     <  Browse   >
 7 │ Probable  3: pro     │    Inference   │8: mpop2..│
 8 │                      │                                     <Qualifiers >
 9 │ Likely    4: lik     │
10 └──────────────────────┘    Confidence  │7: mpop...│         <  Goals    >
11
12                                                              <ConcLusion >
13
14      <   Add    > <  Edit    > <  Delete  > <    OK    > <  Cancel  >
```

Window name: W_kb
Coordinates: FROM 0,0 TO 0,74
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	mbuttons	Push button	"@*HN
2:	mbbuttons	Push button	"@*VN
3:	m.areaname	Field	
4:	m.threshold	Field	
5:	m.probable	Field	
6:	m.likely	Field	
7:	m.rules	Field	
8:	m.isdiag	Check box	"@*C DIAGNOSIS"
9:	m.mpop	Popup	"@^ BAYES;ADDITIVE"
10:	m.mpop2	Popup	"@^ FORWARD;BACKWARD"
11:	mbuttons	Push button	"@*HT
12:	m.threshold	Field	
13:	m.probable	Field	
14:	m.likely	Field	
15:	m.name	Field	
16:	m.rules	Field	
17:	m.mpop	Popup	"@^ BAYES;ADDITIVE"
18:	m.mpop2	Popup	"@^ FORWARD;BACKWARD"
19:	m.isdiag	Check box	"@*C DIAGNOSIS"
20:	minvbutton	Push button	"@*HN \<Knowledge Base"
21:	mbbuttons	Push button	"@*VN

D. QUAL.SCX Last updated: 10/03/94 at 15:01

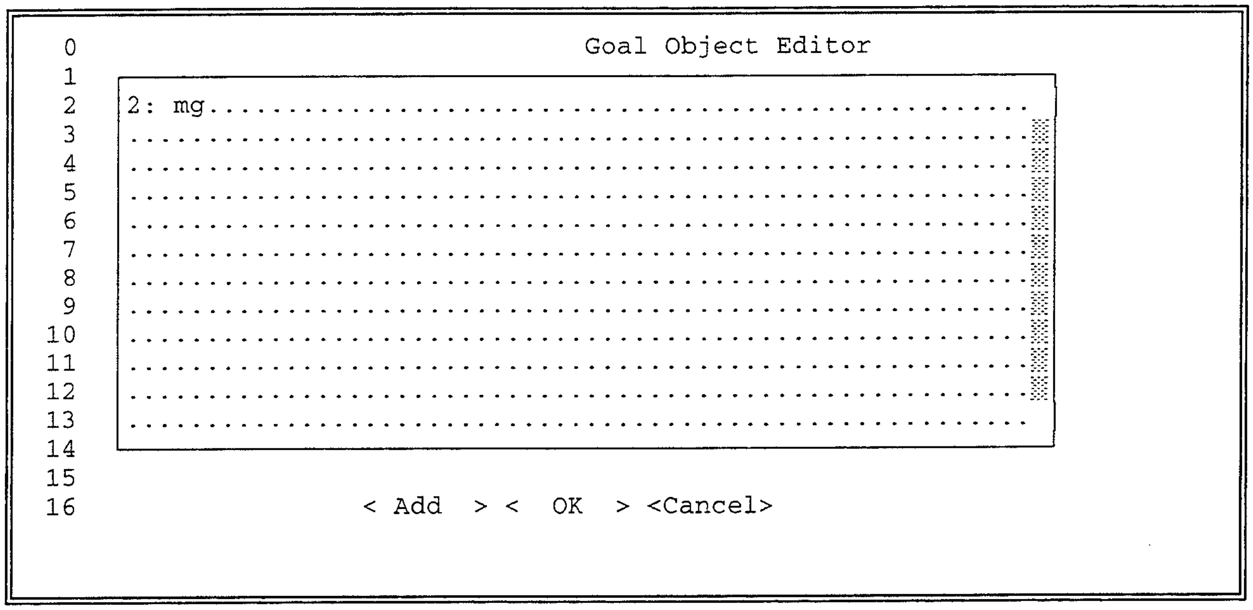


```
                                   Qualifier Editor
 0
 1     2: mq.........................................
 2     ..............................................
 3     ..............................................
 4     ..............................................
 5     ..............................................
 6     ..............................................
 7     ..............................................
 8     ..............................................
 9     ..............................................
10     ..............................................
11     ..............................................
12
13
14             < Add   > <Delete> <  OK   > <Cancel>
```

Window name: W_qe
Coordinates: FROM 0,0 TO 12,57
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	mbuttons	Push button	"@*HT \<Edit;\<Quit"
2:	mq	List	"@&N"
3:	mbuttons	Push button	"@*HN
4:	mq	List	"@&N"

E. GOAL.SCX **Last updated: 10/03/94 at 15:01**

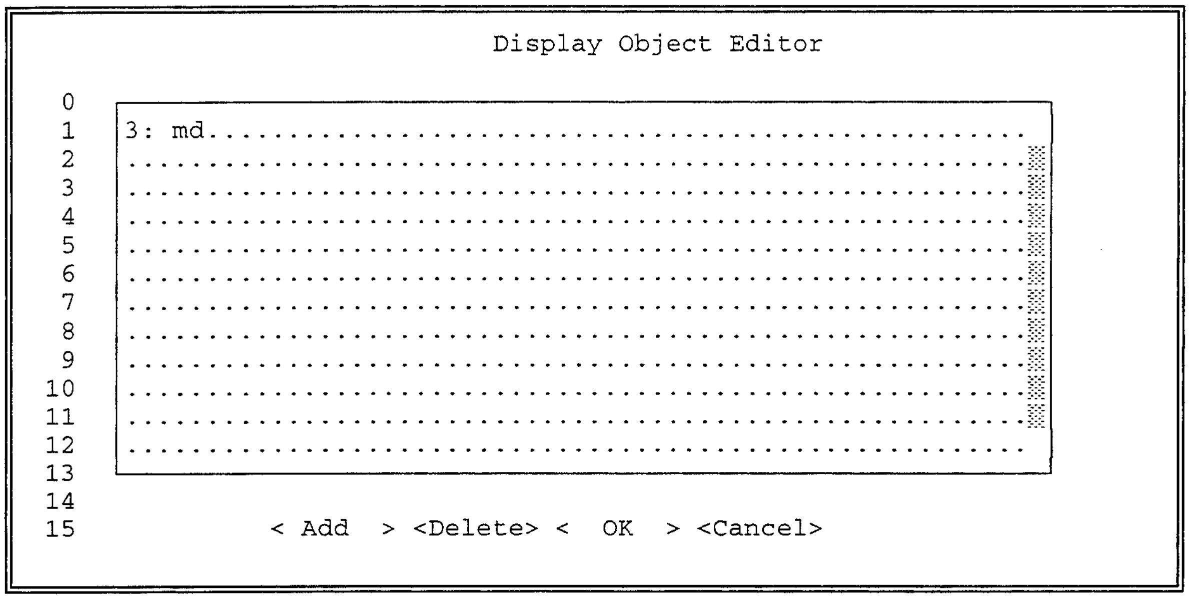


```
 0                     Goal Object Editor
 1   ┌─────────────────────────────────────────────────────────────┐
 2   │2: mg........................................................│
 3   │.............................................................│
 4   │.............................................................│
 5   │.............................................................│
 6   │.............................................................│
 7   │.............................................................│
 8   │.............................................................│
 9   │.............................................................│
10   │.............................................................│
11   │.............................................................│
12   │.............................................................│
13   │.............................................................│
14   └─────────────────────────────────────────────────────────────┘
15
16              < Add   > <  OK   > <Cancel>
```

Window name: W_goal
Coordinates: FROM 0,0 TO 13,63
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: mbuttons	Push button	"@*HN \<Edit;\<Quit"
2: mg	List	"@&N"
3: mbuttons	Push button	"@*HN \<Add;\<OK;\<Cancel"
4: mg	List	"@&N"

F. DISPLAY.SCX Last updated: 10/03/94 at 15:01



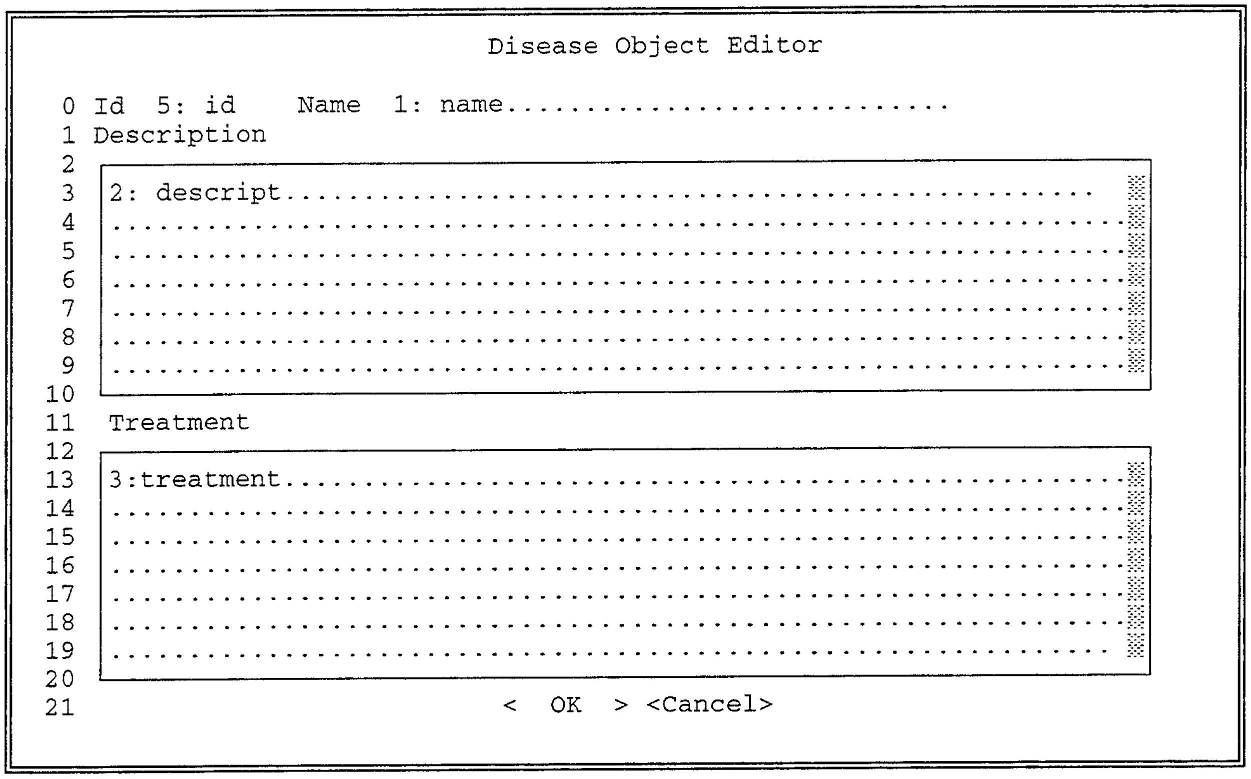
```
                         Display Object Editor

 0    
 1    3: md...........................................................
 2    ................................................................
 3    ................................................................
 4    ................................................................
 5    ................................................................
 6    ................................................................
 7    ................................................................
 8    ................................................................
 9    ................................................................
10    ................................................................
11    ................................................................
12    ................................................................
13
14
15           < Add   > <Delete> <  OK   > <Cancel>
```

Window name: W_displ
Coordinates: FROM 0,0 TO 13,63
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	mbuttons	Push button	"@*HT \<Edit;\<Quit"
2:	md	List	"@&N"
3:	mbuttons	Push button	"@*HT OK;Cancel"
4:	mbbutton	Push button	"@*HN Add;Delete"
5:	md	List	"@&N"

G. DISEASE.SCX Last updated: 10/03/94 at 15:01



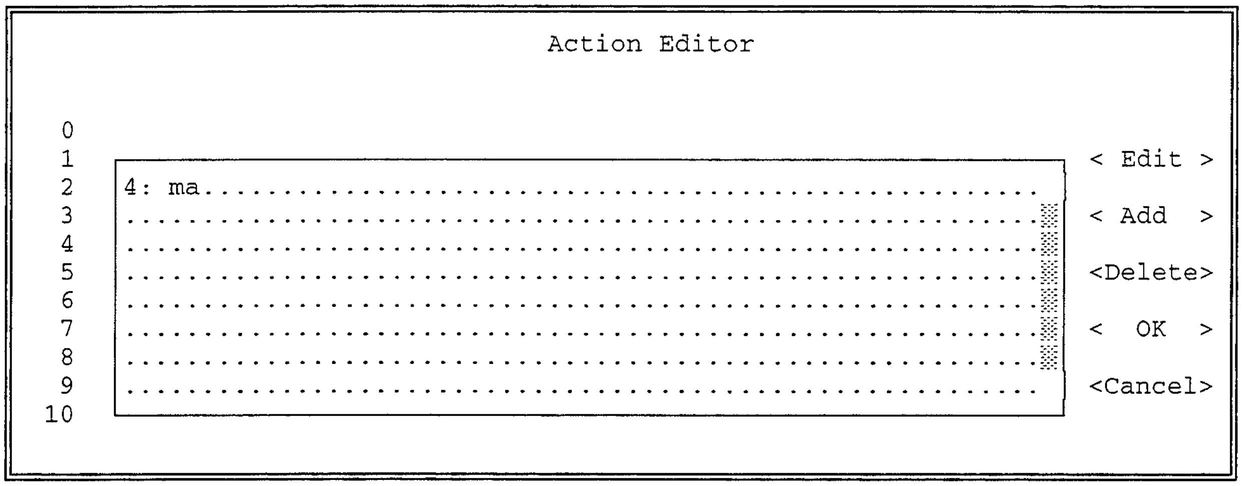
```
                         Disease Object Editor

 0 Id  5: id    Name  1: name...........................
 1 Description
 2
 3  2: descript.................................................... 
 4  ..............................................................
 5  ..............................................................
 6  ..............................................................
 7  ..............................................................
 8  ..............................................................
 9  ..............................................................
10
11  Treatment
12
13  3:treatment...................................................
14  ..............................................................
15  ..............................................................
16  ..............................................................
17  ..............................................................
18  ..............................................................
19  .............................................................. 
20
21                          <  OK  > <Cancel>
```

Window name: W_disease
Coordinates: FROM 0,0 TO 0,77
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	m.id	Field	
2:	m.name	Field	"@!"
3:	m.descript	Field	
4:	m.treatment	Field	
5:	mbuttons	Push button	"@*HT \<OK;\<Cancel"
6:	disease.name	Field	"@!"
7:	disease.descript	Field	
8:	disease.treatment	Field	
9:	mbuttons	Push button	"@*HN \<OK;\<Cancel"
10:	disease.id	Field	

H. ACTION.SCX Last updated: 10/03/94 at 15:01



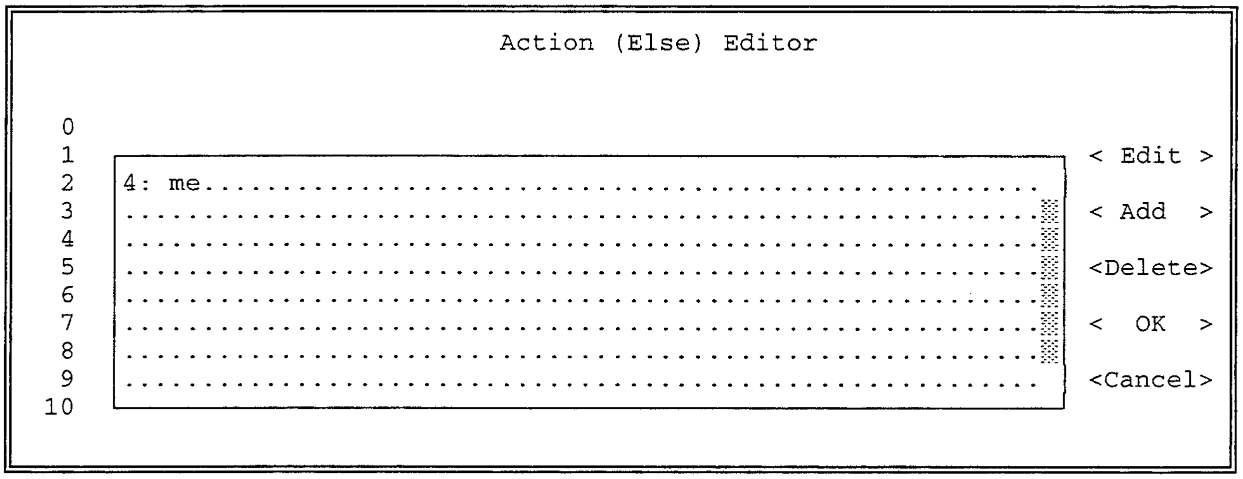
```
                         Action Editor

 0
 1   ┌────────────────────────────────────────────────────┐  < Edit >
 2   │4: ma...............................................│
 3   │...................................................░│  < Add  >
 4   │...................................................░│
 5   │...................................................░│  <Delete>
 6   │...................................................░│
 7   │...................................................░│  <  OK  >
 8   │...................................................░│
 9   │....................................................│  <Cancel>
10   └────────────────────────────────────────────────────┘
```

Window name: W_act
Coordinates: FROM 0,0 TO 9,75
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	mbuttons	Push button	"@*VN \<Edit;\<Quit"
2:	ma	List	"@&N"
3:	mbuttons	Push button	"@*VT \<Delete;\<OK>"
4:	mebutton	Push button	"@*VN \<Edit"
5:	minsbutton	Push button	"@*VN \<Add"
6:	ma	List	"@&N"

I. ACTELSE.SCX Last updated: 10/03/94 at 15:01



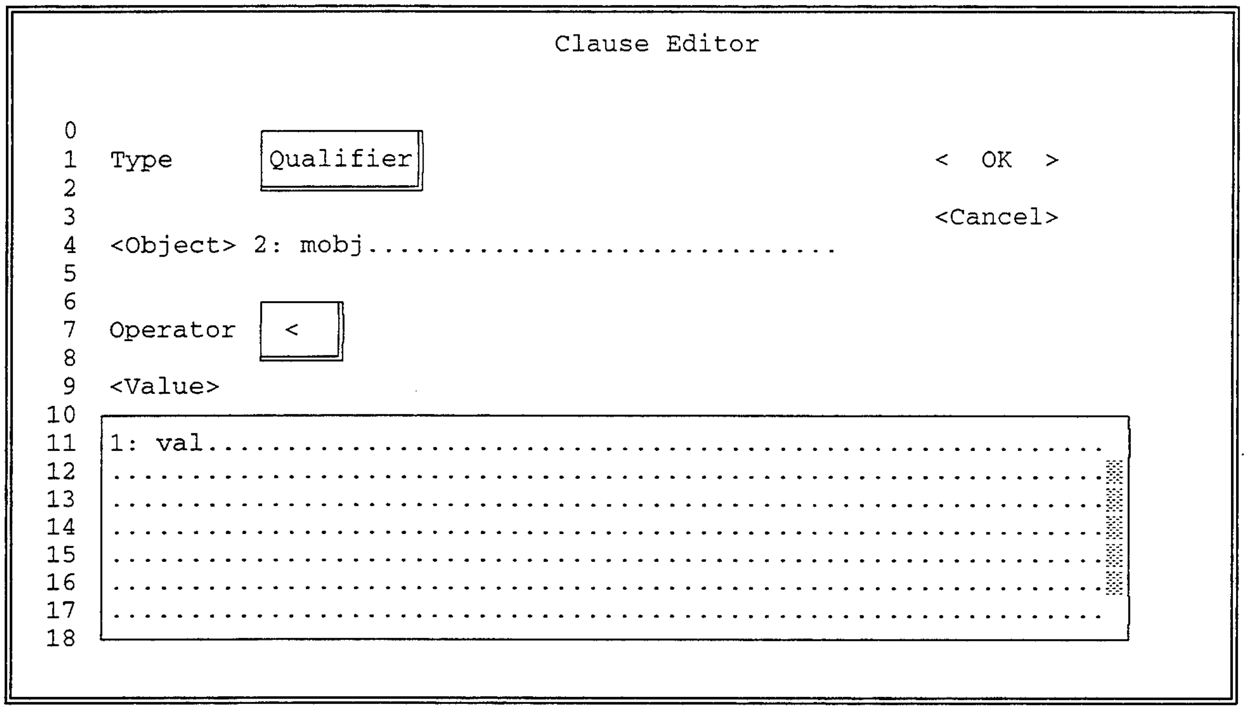
```
                      Action (Else) Editor

 0
 1   ┌────────────────────────────────────────────────────┐  < Edit >
 2   │4: me...............................................│
 3   │...................................................░│  < Add  >
 4   │...................................................░│
 5   │...................................................░│  <Delete>
 6   │...................................................░│
 7   │...................................................░│  <  OK  >
 8   │...................................................░│
 9   │....................................................│  <Cancel>
10   └────────────────────────────────────────────────────┘
```

Window name: W_act
Coordinates: FROM 0,0 TO 9,75
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	mbuttons	Push button	"@*VN \<Edit;\<Quit"
2:	me	List	"@&N"
3:	mbuttons	Push button	"@*VT \<Delete;\<OK>"
4:	mebutton	Push button	"@*VN \<Edit"
5:	minsbutton	Push button	"@*VN \<Add"
6:	me	List	"@&N"

J. CLAUSE.SCX Last updated: 10/03/94 at 15:01



```
                              Clause Editor

Type      [Qualifier]                                    <  OK  >

                                                         <Cancel>
<Object> 2: mobj..............................


Operator  [ < ]

<Value>

1: val..............................................................
....................................................................
....................................................................
....................................................................
....................................................................
....................................................................
....................................................................
```

Window name: W_clause
Coordinates: FROM 0,0 TO 0,67
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	mbuttons	Push button	"@*VT \<OK;\<Cancel"
2:	mtype	Popup	"@^ Qualifier;Goal"
3:	mobj	Field	
4:	m.op	Popup	"@^ <;<=;==;>=;>;!=;="
5:	m.val	Field	
6:	m.val	Field	
7:	mobj	Field	
8:	m.op	Popup	"@^ <;<=;==;>=;>;!=;="
9:	mtype	Popup	"@^ Qualifier;Goal"
10:	mbuttons	Push button	"@*VT \<OK;\<Cancel"

K. RULE.SCX **Last updated: 10/03/94 at 15:01**

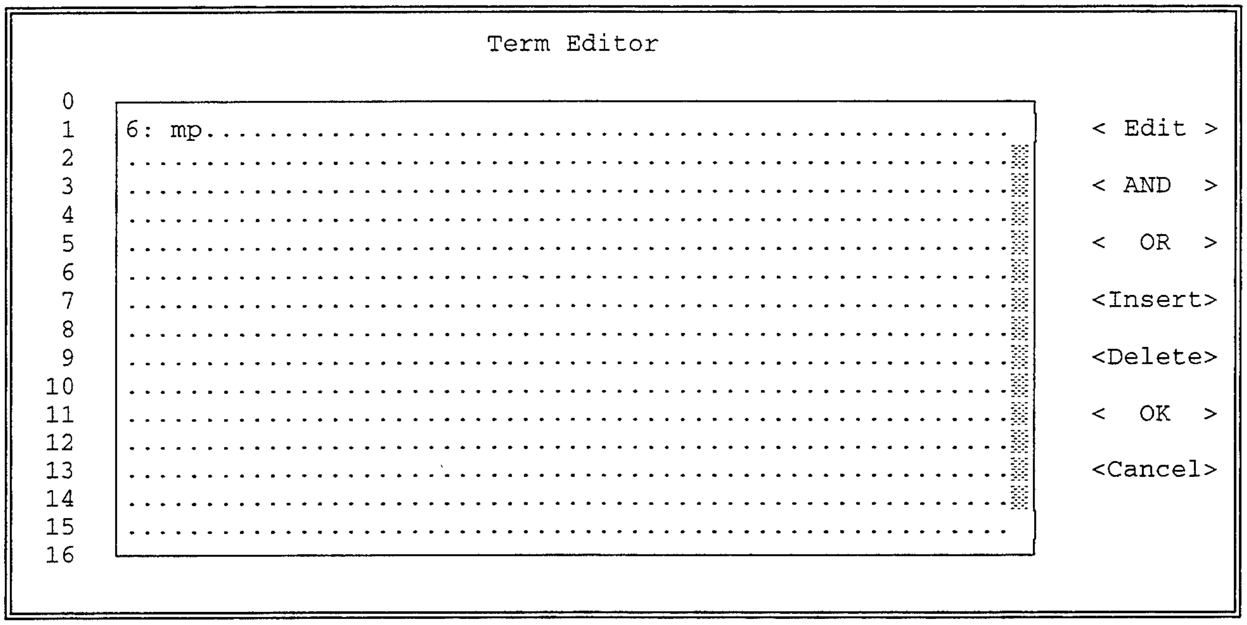
```
                              Rule Object Editor


 0
 1  Rule 1: ru    Salience 2: s
 2
 3  Explanation
 4 ┌──────────────────────────────────────────────────────────────────┐
 5 │5: explain.........................................................░│
 6 │...................................................................░│
 7 │...................................................................░│
 8 │...................................................................░│
 9 │...................................................................░│
10 └──────────────────────────────────────────────────────────────────┘
11  Note
12 ┌──────────────────────────────────────────────────────────────────┐
13 │6: note............................................................░│
14 │...................................................................░│
15 │...................................................................░│
16 │...................................................................░│
17 │...................................................................░│
18 └──────────────────────────────────────────────────────────────────┘
19                  < Add   > <Delete > <  Ok    > <Cancel >
```

Window name: W_rule Window name: W_rule
Coordinates: FROM 0,0 TO 5,75
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	m.rule	Field	
2:	m.salience	Field	
3:	mbuttons	Push button	"@*HT \<OK ;\<Cancel"
4:	m.explain	Field	
5:	m.note	Field	
6:	m.rule	Field	
7:	m.salience	Field	
8:	mbbutton	Push button	"@*VN \<Add"
9:	mbuttons	Push button	"@*HT
10:	m.explain	Field	
11:	m.note	Field	

L. TERM.SCX **Last updated: 10/03/94 at 15:01**



```
                          Term Editor

 0
 1  6: mp.............................................    < Edit >
 2  ..................................................
 3  ..................................................    < AND  >
 4  ..................................................
 5  ..................................................    <  OR  >
 6  ..................................................
 7  ..................................................    <Insert>
 8  ..................................................
 9  ..................................................    <Delete>
10  ..................................................
11  ..................................................    <  OK  >
12  ..................................................
13  ..................................................    <Cancel>
14  ..................................................
15  ..................................................
16
```

Window name: W_term
Coordinates: FROM 0,0 TO 16,75
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	mp	List	"@&N"
2:	mbuttons	Push button	"@*VN \<Edit;\<Quit"
3:	mbuttons	Push button	"@*VT \<Del
4:	mbbutton	Push button	"@*VN \<AND"
5:	morbutton	Push button	"@*VN \<OR"
6:	mebutton	Push button	"@*VN \<Edit"
7:	minsbutton	Push button	"@*VN \<Insert"
8:	mp	List	"@&N"

M. OBJECT.SCX Last updated: 10/03/94 at 15:01

```
                          Add New Object


0
1
2  Type:  (■) Subject     ( ) Disease
3
4  <Object> 2: name..............................
5
6  ID# :  3: id
7
8                    <  Ok  > <Cancel>
```

Window name: Object
Coordinates: FROM 0,0 TO 0,53
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	m.object	Radio button	"@*RHN Subject;Disease"
2:	m.name	Field	
3:	m.id	Field	
4:	m.button	Push button	"@*HT \<OK;\<Cancel"
5:	m.obj	Push button	"@*HN Object"
6:	m.object	Radio button	"@*RHN Subject;Disease"
7:	m.name	Field	
8:	m.id	Field	
9:	m.button	Push button	"@*HT \<OK;\<Cancel"
10:	m.obj	Push button	"@*HN Object"

N. ENUM.SCX Last updated: 10/03/94 at 15:01

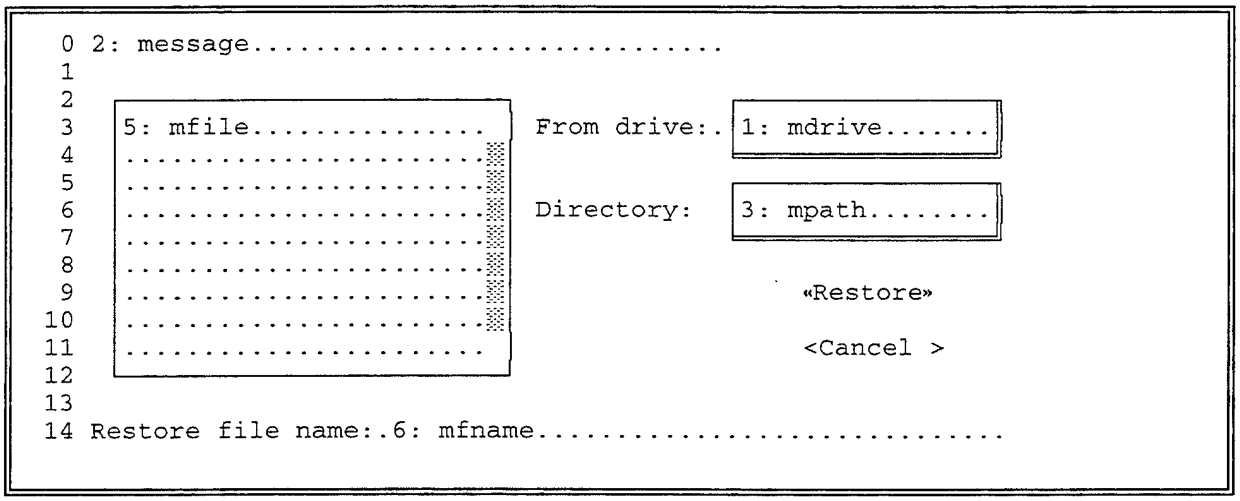
```
                       Enumerated Type Editor


0 Mutex 3
1 Enum  4: enumerate..........................................
2 Ord   2:
3                   < Add  >  <  OK  > <Cancel>
```

Window name: W_qenum
Coordinates: FROM 0,0 TO 0,70
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	mbuttons	Push button	"@*HT OK;Cancel"
2:	enum.ord	Field	
3:	enum.mutex	Field	
4:	enum.enumerate	Field	"@!"
5:	mbbutton	Push button	"@*VN Add"
6:	mbuttons	Push button	"@*HT OK;Cancel"
7:	enum.ord	Field	
8:	enum.mutex	Field	
9:	enum.enumerate	Field	"@!"
10:	mbbutton	Push button	"@*VN Add"

O. RESTORE.SCX Last updated: 10/03/94 at 15:01



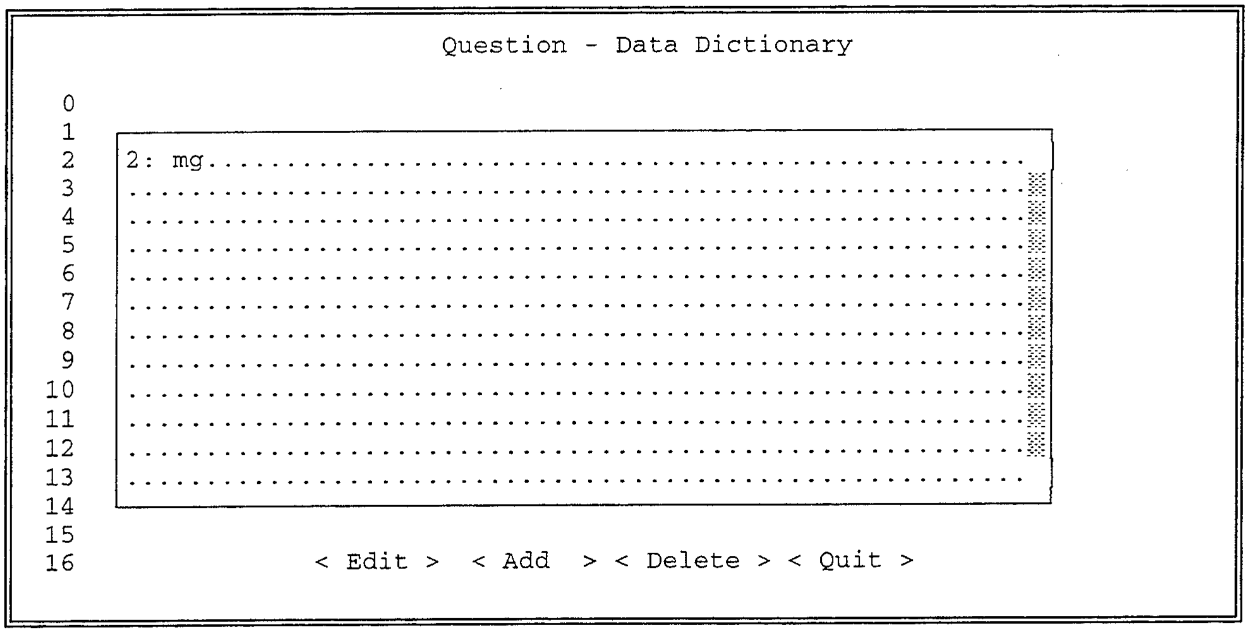
```
 0 2: message.............................
 1
 2
 3   5: mfile...............    From drive:. 1: mdrive.......
 4   .......................
 5   .......................
 6   .......................    Directory:   3: mpath........
 7   .......................
 8   .......................
 9   .......................                   «Restore»
10   .......................
11   .......................                   <Cancel >
12
13
14 Restore file name:.6: mfname..............................
```

Window name: W_restore
Coordinates: FROM 0,0 TO 0,60
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	mdrive	Popup	"@^ "
2:	message	Field	
3:	mpath	Popup	"@^ "
4:	maction	Push button	"@*HT \!\<Restore;\<Cancel"
5:	mfile	List	"@&N"
6:	mfname	Field	
7:	mdrive	Popup	"@^ "
8:	message	Field	
9:	mpath	Popup	"@^ "
10:	maction	Push button	"@*VT \!\<Restore;\<Cancel"

11: mfile	List	"@&N"
12: mfname	Field	

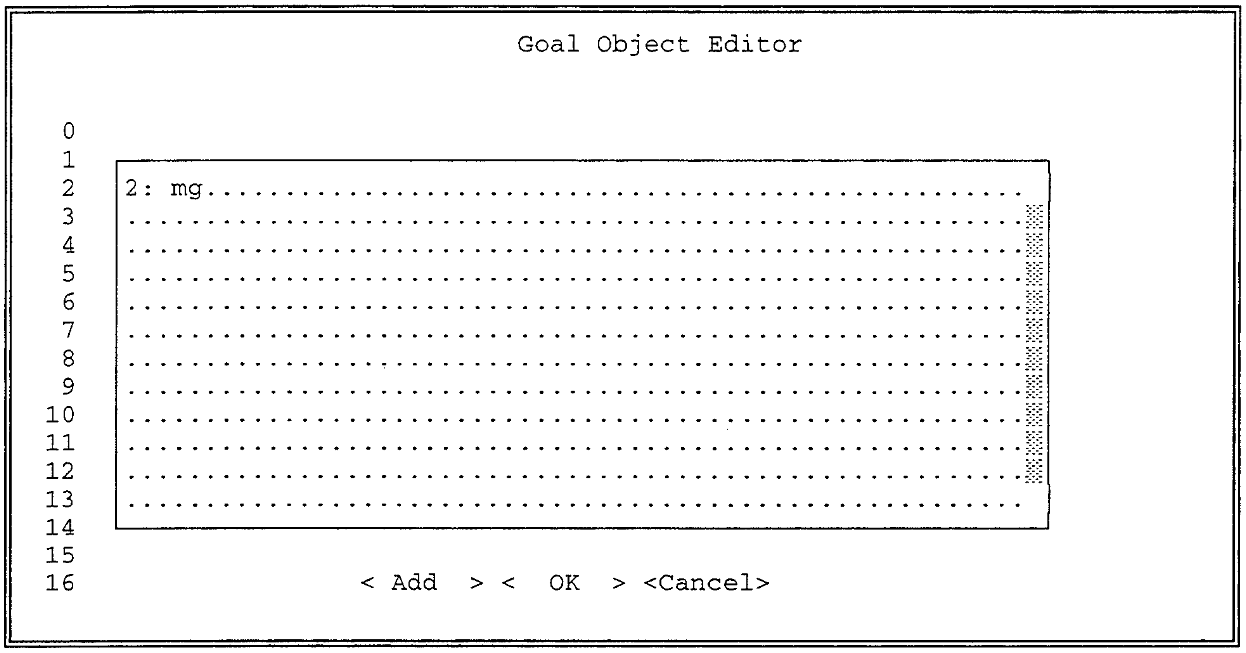
P. DD.SCX **Last updated: 10/03/94 at 15:01**


Question - Data Dictionary
2: mg
< Edit > < Add > < Delete > < Quit >

```
Window name: W_
Coordinates: FROM 0,0 TO 0,60
Window options: FLOAT CLOSE MINIMIZE SHADOW
```

Name	Type	Picture
1: mbuttons	Push button	"@*HN \<Edit;\<Add;
2: mq	List	"@&N"

Q. PLAN.SCX **Last updated: 10/03/94 at 15:01**

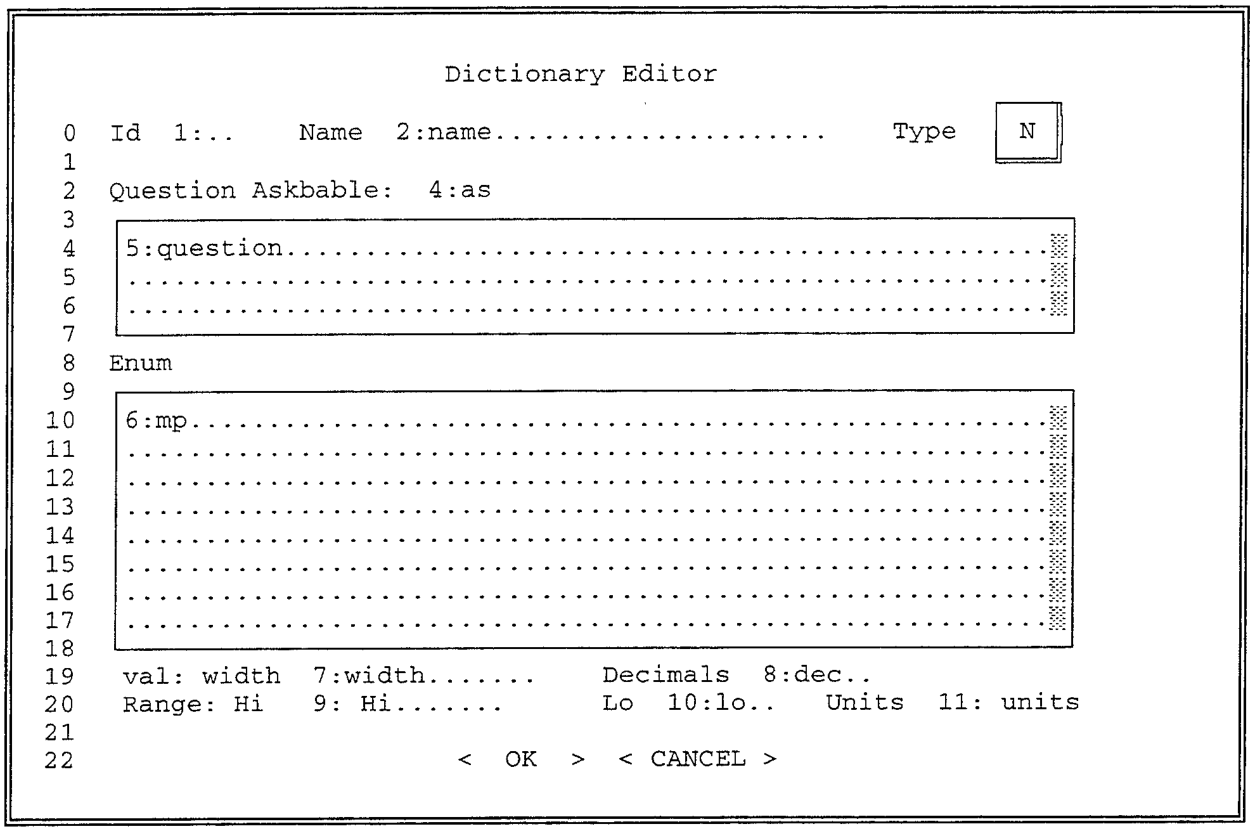


```
                         Goal Object Editor

 0
 1
 2   2: mg.........................................................
 3   .............................................................
 4   .............................................................
 5   .............................................................
 6   .............................................................
 7   .............................................................
 8   .............................................................
 9   .............................................................
10   .............................................................
11   .............................................................
12   .............................................................
13   .............................................................
14
15
16                < Add  > <  OK  > <Cancel>
```

Window name: W_goal
Coordinates: FROM 0,0 TO 13,63
Window options: FLOAT CLOSE MINIMIZE SHADOW

	Name	Type	Picture
1:	mbuttons	Push button	"@*HN \<Add;\<OK;\<Cancel"
2:	mg	List	"@&N"
3:	mbuttons	Push button	"@*HN \<Add;\<OK;\<Cancel"
4:	mg	List	"@&N"



```
                         Dictionary Editor
 0  Id  1:..    Name  2:name....................    Type  [ N ]
 1
 2  Question Askbable:  4:as
 3
 4   5:question...............................................
 5   .........................................................
 6   .........................................................
 7
 8  Enum
 9
10   6:mp.....................................................
11   .........................................................
12   .........................................................
13   .........................................................
14   .........................................................
15   .........................................................
16   .........................................................
17   .........................................................
18
19   val: width  7:width.......    Decimals  8:dec..
20   Range: Hi   9: Hi.......      Lo  10:lo..   Units  11: units
21
22                         <  OK  >  < CANCEL >
```

	Name	Type	Picture
1:	m.id	Field	
2:	m.name	Field	
3:	m.datatype	Popup	"@^ N;E;M;L"
4:	dict.askable	Field	
5:	m.question	Field	
6:	mp	List	"@&N"
7:	m.width	Field	
8:	m.dec	Field	
9:	m.hi	Field	
10:	m.lo	Field	
11:	m.units	Field	
12:	mbuttons	Push button	"@*HT OK;Cancel"

S. KBEDIT.SCX Last updated: 10/03/94 at 15:01

```
                        Knowledge Base Rule Editor
 0  Knowledge Base:  1:areaname....................       <  Next   >
 1  Rule:  2:Rule..........
 2  < IF: >                                               <  Prev   >
 3   ____________________________________________________
 4  | 4: mp..............................................|  < Browse >
 5  | ...................................................|
 6  | ...................................................|  <  Edit   >
 7   ----------------------------------------------------
 8  < THEN: >                                             <  eXit   >
 9   ____________________________________________________
10  | 6: ma..............................................|
11  | ...................................................|
12  | ...................................................|
13   ----------------------------------------------------
14  < ELSE: >
15   ____________________________________________________
16  | 8: me..............................................|
17  | ...................................................|
18  | ...................................................|
19   ----------------------------------------------------
20   val: width  8: width.......   Decimals  9: dec..
21   Range: Hi   10: Hi.......     Lo  11:lo..   Units  12: units
22
23                          <  OK  >  < CANCEL >
```

Name	Type	Picture
1: m.areaname	Field	
2: rule.rule	Field	
3: minvprem	Push button	"@*HN \<IF:"
4: mp	List	"@&N"
5: minvact	Push button	"@*HN \<THEN:"
6: ma	List	"@&N"
7: minvelse	Push button	"@*HN \<ELSE:"
8: me	List	"@&N"
9: mbbutton	Push button	"@*VN

T. OBLIST.SCX **Last updated: 10/03/94 at 15:01**

```
                        Object list

 0
 1   1: obj..............................................
 2   ....................................................
 3   ....................................................
 4   ....................................................
 5   ....................................................
 6   ....................................................
 7   ....................................................
 8   ....................................................
 9   ....................................................
10   ....................................................
11   ....................................................
12   ....................................................
13   ....................................................
14   ....................................................
15   ....................................................
16
17
18                  < New  > <  Ok  > <Cancel>
```

Window name: W_obj
Coordinates: FROM 0,0 TO 0,61
Window options: FLOAT CLOSE MINIMIZE SHADOW

Name	Type	Picture
1: m.obj	List	"@&N"
2: m.buttons	Push button	"@*HT \<New;\<Ok;\<Cancel"

U. DDEDIT.SCX Last updated: 10/03/94 at 15:01

```
                              Data Element Editor

 0  Id  1:..    Name  2:name....................    Type  | N |
 1
 2  Use in rule
 3
 4   4. rulesuse............................................
 5   .......................................................
 6   .......................................................
 7
 8  Question Askbable:  5:as
 9
10   6. question............................................
11   .......................................................
12   .......................................................
13
14  Enum
15
16   7. mp..................................................
17   .......................................................
18   .......................................................
19
20   val: width   8: width.......   Decimals  9: dec..
21   Range: Hi    10: Hi.......     Lo  11:lo..   Units   12: units
22
23                              <  OK  >  < CANCEL >
```

	Name	Type	Picture
1:	m.id	Field	
2:	m.name	Field	
3:	m.datatype	Popup	"@^ N;E;M;L"
4:	m.rulesuse	List	"@&N"
5:	m.askable	Field	
6:	m.question	Field	
7:	mp	List	"@&N"
8:	m.width	Field	
9:	m.dec	Field	
10:	m.hi	Field	
11:	m.lo	Field	
12:	m.units	Field	
13:	mbuttons	Push button	"@*HT OK;Cancel"

V. DDENUM.SCX Last updated: 10/03/94 at 15:01

```
0 Mutex 2:
1 Enum  3: enumerate...........................................................
2 Ord   1:
3                      < Add  >  <  OK  > <Cancel>
```

	Name	Type	Picture
1:	m.ord	Field	
2:	m.mutex	Field	
3:	m.enumerate	Field	"@!"
4:	mbuttons	Push button	"@*HN Add;OK;Cancel"

W. KBLOAD.SCX Last updated: 10/03/94 at 15:01

```
 0
 1  Read From Definition file:
 2  1: src......................
 3
 4
 5  Create Files in Temporary directory:
 6  2: new......................
 7  ┌────────────────────────┐
 8  │3: dbf..................│
 9  │........................░
10  │........................░
11  │........................░
12  │........................░
13  │........................░       < Load >
14  │........................│
15  └────────────────────────┘       <Cancel>
```

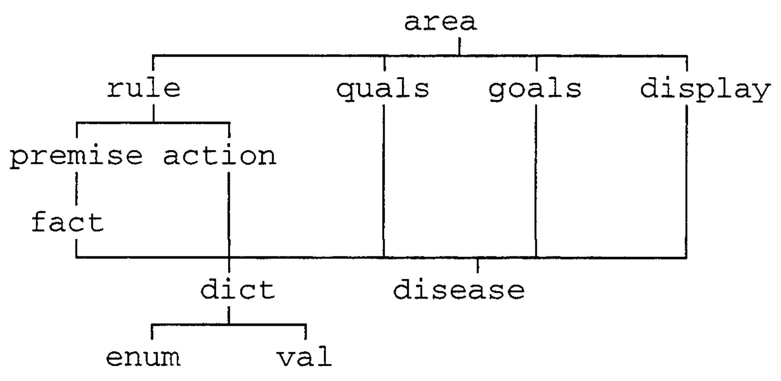
	Name	Type	Picture
1:	m.src	Field	
2:	m.new	Field	
3:	m.dbf	List	"@&N"
4:	m.load	Push button	"@*VN Load"
5:	mbuttonc	Push button	"@*VT Cancel"
6:	m.src	Field	
7:	m.new	Field	
8:	m.dbf	List	"@&N"
9:	m.load	Push button	"@*VT Load"
10:	mbuttonc	Push button	"@*VT Cancel"

Section IV. Data Dictionary. The system contains 14 databases.

DICT.DBF	-- Data element dictionary
ACTION.DBF	-- List of action for each rule
DISEASE.DBF	-- Disease element dictionary
ENUM.DBF	-- List of values for enumerate types
HELP.DBF	-- Fox command help text
KBHELP.DBF	-- Knowledge Base Command help text
PREMISE.DBF	-- List of premises for each rule
RULE.DBF	-- List of qualifiers for each knowledge base
VAL.DBF	-- Attributes for each numeric type
AREA.DBF	-- Diagnostic area or knowledge base
GOALS.DBF	-- List of goals for each knowledge base
DISPLAY.DBF	-- List of qualifiers shown for each knowledge base
QUALS.DBF	-- List of qualifiers for each knowledge base
FACT.DBF	-- Factors in rule premises

A. Knowledge Base Structure (relationship of database)

The following chart shows the structure of the knowledge base. This structure is used to set up relationships between the database and knowledge base:



Important facts about the knowledge base include:

The **AREA** file is the primary parent of all other files, as it relates to the inference engine. Therefore, selecting an area or knowledge base, means that the associated set of rules, qualifiers, goals and displays is also selected.

The **RULE** file is the parent of the PREMISE and ACTION files. Therefore, selecting a RULE, means that a set of PREMISE and ACTION clauses is also selected for that rule. The inference engine can select a particular rule and then retrieve corresponding PREMISE and ACTION clauses from the child files. Since the RULE:PREMISE and RULE:ACTION relationships are one-to-many, several PREMISE and ACTION clauses can be associated with one rule.

The **PREMISE** file is the parent of the FACT file. Therefore, selecting a premise means that a set of fact clauses is also selected for that premise. When the inference engine evaluates a rule premise, it can retrieve the corresponding FACT clauses from the child file.

The **DICT** and **DISEASE** files are the children files of several of the knowledge base tables, including the QUALS, GOALS, DISPLAY, FACT, and ACTION files. The DICT and DISEASE files therefore play a central role in the knowledge base structure. These are the repositories for the basic data element definitions that are used in many places throughout the knowledge base.

The **DICT** file is the parent of the ENUM and VAL files. Each DICT element can have several ENUM entries associated with it,since this is a one-to-many relationship. Each DICT element can have one VAL entry associated with it, since this is a one-to-one relationship.

B. Database Structure Summary.

1. Structure for table/dbf: **DICT.DBF**
Number of data records : 256 Last updated : 06/17/93

Field	Field name	Type	Width	Dec	Start	End
1	ID	Numeric	4	0	1	4
2	NAME	Character	80	0	5	84
3	ALIAS	Character	20	0	85	104
4	DATATYPE	Character	1	0	105	105
5	ASKABLE	Logical	1	0	106	106
6	QUESTION	Character	100	0	107	206
7	PROMPT	Character	80	0	207	286
8	META	Logical	1	0	287	287
** Total **			288			

This table/dbf is associated with index file/tag(s):
: DICTID.IDX (ID)
: DICTNAME.IDX (NAME)
Used by: KB.SPR and DD.SPR

2. Structure for table/dbf: **DISEASE.DBF**
Number of data records : 51 Last updated : 06/10/93

Field	Field name	Type	Width	Dec	Start	End

Field	Field name	Type	Width	Dec	Start	End
1	ID	Numeric	5	0	1	5
2	NAME	Character	80	0	6	85
3	ALIAS	Character	20	0	86	105
4	DESCRIPT	Memo	10	0	106	115
5	TREATMENT	Memo	10	0	116	125
6	BRIEF	Memo	10	0	126	135
** Total **			136			

This table/dbf is associated with the memo file: DISEASE.FPT
This table/dbf is associated with index file/tag(s):
: DISID.IDX (ID)
: DISEASE.IDX (NAME)
Used by: KB.SPR

3. Structure for table/dbf: **ACTION.DBF**
Number of data records : 2344 Last updated : 01/12/93

Field	Field name	Type	Width	Dec	Start	End
1	CLAUSE	Numeric	5	0	1	5
2	OP	Character	2	0	6	7
3	OBJECT	Character	1	0	8	8
4	ID	Numeric	5	0	9	13
5	TAG	Character	1	0	14	14
6	VAL	Character	10	0	15	24
7	TEXT	Memo	10	0	25	34
** Total **			35			

This table/dbf is associated with the memo file: ACTION.FPT
This table/dbf appears to be associated with index file/tag(s):
: ACTION.IDX (CLAUSE)
Used by: KBLDR.PRG, KBDELETE.PRG, and KBEDIT.SPR

4. Structure for table/dbf: **ENUM.DBF**
Number of data records : 698 Last updated : 06/07/93

Field	Field name	Type	Width	Dec	Start	End
1	ID	Numeric	4	0	1	4
2	ORD	Numeric	2	0	5	6
3	MUTEX	Character	1	0	7	7
4	ENUMERATE	Character	80	0	8	87
5	REPORT	Character	80	0	88	167
** Total **			168			

This table/dbf appears to be associated with index file/tag(s):
: ENUM.IDX (ID)
Used by: KB.SPR and DD.SPR

5. Structure for table/dbf: **HELP.DBF**
Number of data records : 147 Last updated : 09/17/93

Field	Field name	Type	Width	Dec	Start	End

```
    1  TOPIC        Character      80       0         1         80
    2  DETAILS      Memo           10       0        81         90
    3  CLASS        Character      20       0        91        110
    4  ID           Numeric         5       0       111        115
    5  SOURCE       Character       1       0       116        116
** Total **                       117
```

This table/dbf is associated with the memo file: HELP.FPT

6. Structure for table/dbf: **KBHELP.DBF** Alias: HELP
Number of data records : 20 Last updated : 05/29/92

Field	Field name	Type	Width	Dec	Start	End
1	TOPIC	Character	30	0	1	30
2	DETAILS	Memo	10	0	31	40
3	CLASS	Character	20	0	41	60
4	ID	Numeric	5	0	61	65
5	SOURCE	Character	1	0	66	66
** Total **			67			

This table/dbf is associated with the memo file: KBHELP.FPT
Used by: KBLDR.PRG

7. Structure for table/dbf: **PREMISE.DBF**
Number of data records : 1809 Last updated : 01/12/93

Field	Field name	Type	Width	Dec	Start	End
1	CLAUSE	Numeric	5	0	1	5
2	OP	Character	1	0	6	6
3	FACT	Numeric	5	0	7	11
4	FACTR	Numeric	5	0	12	16
** Total **			17			

This table/dbf appears to be associated with index file/tag(s):
: PREMISE.IDX (CLAUSE)
Used by: KBLDR.PRG, KBDELETE.PRG, and KBEDIT.SPR

8. Structure for table/dbf: **RULE.DBF**
Number of data records : 564 Last updated : 06/01/93

Field	Field name	Type	Width	Dec	Start	End
1	RULE	Numeric	5	0	1	5
2	AREA	Numeric	4	0	6	9
3	SALIENCE	Numeric	3	0	10	12
4	PREMISE	Numeric	5	0	13	17
5	ACTION	Numeric	5	0	18	22
6	ELSE	Numeric	5	0	23	27
7	QUALS	Memo	10	0	28	37
8	GOALS	Memo	10	0	38	47
9	NOTE	Memo	10	0	48	57
10	EXPLAIN	Memo	10	0	58	67
** Total **			68			

This table/dbf is associated with the memo file: RULE.FPT
This table/dbf appears to be associated with index file/tag(s):
: RULEAREA.IDX (AREA)
: SALIENCE.IDX (SALIENCE)
: RULE.IDX (RULE)
Used by: KBLDR.PRG, KBDELETE.PRG, and KBEDIT.SPR

9. Structure for table/dbf: **VAL.DBF**
Number of data records : 19 Last updated : 04/29/93

Field	Field name	Type	Width	Dec	Start	End
1	ID	Numeric	4	0	1	4
2	WIDTH	Numeric	2	0	5	6
3	DEC	Numeric	1	0	7	7
4	LO	Numeric	9	2	8	16
5	HI	Numeric	9	2	17	25
6	UNITS	Character	10	0	26	35
** Total **			36			

This table/dbf appears to be associated with index file/tag(s):
: VAL.IDX (ID)
Used by: KB.SPR and DD.SPR

10. Structure for table/dbf: **AREA.DBF**
Number of data records : 4 Last updated : 06/17/93

Field	Field name	Type	Width	Dec	Start	End
1	AREA	Numeric	4	0	1	4
2	NAME	Character	30	0	5	34
3	INFERENCE	Numeric	1	0	35	35
4	METHOD	Numeric	1	0	36	36
5	ISDIAG	Numeric	1	0	37	37
6	SIGNON	Character	30	0	38	67
7	START	Character	30	0	68	97
8	FINISH	Character	30	0	98	127
9	THRESHOLD	Numeric	6	2	128	133
10	PROBABLE	Numeric	6	2	134	139
11	LIKELY	Numeric	6	2	140	145
12	RULES	Numeric	4	0	146	149
** Total **			150			

This table/dbf appears to be associated with index file/tag(s):
: AREA.IDX (AREA)
Used by: KB.SPR and DD.SPR

11. Structure for table/dbf: **GOALS.DBF**
Number of data records : 89 Last updated : 05/11/93

Field	Field name	Type	Width	Dec	Start	End
1	AREA	Numeric	4	0	1	4
2	OBJECT	Character	1	0	5	5
3	ID	Numeric	5	0	6	10
** Total **			11			

This table/dbf appears to be associated with index file/tag(s):
: GOALS.IDX (ID)
Used by: KB.SPR and DD.SPR

12. Structure for table/dbf: **DISPLAY.DBF**
Number of data records : 5 Last updated : 12/07/92

Field	Field name	Type	Width	Dec	Start	End
1	AREA	Numeric	5	0	1	5
2	ID	Numeric	5	0	6	10
3	OBJECT	Character	1	0	11	11
** Total **			12			

This table/dbf is not associated with index files/tags(s).
Used by: KB.SPR

13. Structure for table/dbf: **QUALS.DBF**
Number of data records : 292 Last updated : 01/12/93

Field	Field name	Type	Width	Dec	Start	End
1	AREA	Numeric	4	0	1	4
2	OBJECT	Character	1	0	5	5
3	ID	Numeric	5	0	6	10
4	RULES	Memo	10	0	11	20
5	RULESO	Memo	10	0	21	30
** Total **			31			

This table/dbf is associated with the memo file: QUALS.FPT
This table/dbf appears to be associated with index file/tag(s):
: QUALS.IDX (ID)
Used by: KB.SPR and DD.SPR

14. Structure for table/dbf: **FACT.DBF**
Number of data records : 1223 Last updated : 01/12/93

Field	Field name	Type	Width	Dec	Start	End
1	CLAUSE	Numeric	5	0	1	5
2	OP	Character	2	0	6	7
3	OBJECT	Character	1	0	8	8
4	ID	Numeric	5	0	9	13
5	TAG	Character	1	0	14	14
6	VAL	Character	10	0	15	24
7	TEXT	Memo	10	0	25	34
** Total **			35			

This table/dbf is associated with the memo file: FACT.FPT
This table/dbf appears to be associated with index file/tag(s):
: FACT.IDX (CLAUSE)
Used by: KBLDR.PRG, KBDELETE.PRG, and KBEDIT.SPR

C. Database Field Summary

Field Name	Type	Len	Dec	Table/DBF
ACTION	N	5	0	RULE.DBF
ALIAS	C	20	0	DICT.DBF
ALIAS	C	20	0	DISEASE.DBF
AREA	N	4	0	AREA.DBF
AREA	N	5	0	DISPLAY.DBF
AREA	N	4	0	GOALS.DBF
AREA	N	4	0	QUALS.DBF
AREA	N	4	0	RULE.DBF
ASKABLE	L	1	0	DICT.DBF
BRIEF	M	10	0	DISEASE.DBF
CLASS	C	20	0	HELP.DBF
CLASS	C	20	0	KBHELP.DBF
CLAUSE	N	5	0	ACTION.DBF
CLAUSE	N	5	0	FACT.DBF
CLAUSE	N	5	0	PREMISE.DBF
DATATYPE	C	1	0	DICT.DBF
DEC	N	1	0	VAL.DBF
DESCRIPT	M	10	0	DISEASE.DBF
DETAILS	M	10	0	HELP.DBF
DETAILS	M	10	0	KBHELP.DBF
ELSE	N	5	0	RULE.DBF
ENUMERATE	C	80	0	ENUM.DBF
EXPLAIN	M	10	0	RULE.DBF
FACT	N	5	0	PREMISE.DBF
FACTR	N	5	0	PREMISE.DBF
FINISH	C	30	0	AREA.DBF
GOALS	M	10	0	RULE.DBF
HI	N	9	2	VAL.DBF
ID	N	5	0	ACTION.DBF
ID	N	4	0	DICT.DBF
ID	N	5	0	DISEASE.DBF
ID	N	5	0	DISPLAY.DBF
ID	N	4	0	ENUM.DBF
ID	N	5	0	FACT.DBF
ID	N	5	0	GOALS.DBF
ID	N	5	0	HELP.DBF
ID	N	5	0	KBHELP.DBF
ID	N	5	0	QUALS.DBF
ID	N	4	0	VAL.DBF
INFERENCE	N	1	0	AREA.DBF
ISDIAG	N	1	0	AREA.DBF
LIKELY	N	6	2	AREA.DBF
LO	N	9	2	VAL.DBF

META	L	1	0	DICT.DBF
METHOD	N	1	0	AREA.DBF
MUTEX	C	1	0	ENUM.DBF

Field Name	Type	Len	Dec	Table/DBF
NAME	C	30	0	AREA.DBF
NAME	C	80	0	DICT.DBF
NAME	C	80	0	DISEASE.DBF
NOTE	M	10	0	RULE.DBF
OBJECT	C	1	0	ACTION.DBF
OBJECT	C	1	0	DISPLAY.DBF
OBJECT	C	1	0	FACT.DBF
OBJECT	C	1	0	GOALS.DBF
OBJECT	C	1	0	QUALS.DBF
OP	C	2	0	ACTION.DBF
OP	C	2	0	FACT.DBF
OP	C	1	0	PREMISE.DBF
ORD	N	2	0	ENUM.DBF
PREMISE	N	5	0	RULE.DBF
PROBABLE	N	6	2	AREA.DBF
PROMPT	C	80	0	DICT.DBF
QUALS	M	10	0	RULE.DBF
QUESTION	C	100	0	DICT.DBF
REPORT	C	80	0	ENUM.DBF
RULE	N	5	0	RULE.DBF
RULES	N	4	0	AREA.DBF
RULES	M	10	0	QUALS.DBF
RULESO	M	10	0	QUALS.DBF
SALIENCE	N	3	0	RULE.DBF
SIGNON	C	30	0	AREA.DBF
SOURCE	C	1	0	HELP.DBF
SOURCE	C	1	0	KBHELP.DBF
START	C	30	0	AREA.DBF
TAG	C	1	0	ACTION.DBF
TAG	C	1	0	FACT.DBF
TEXT	M	10	0	ACTION.DBF
TEXT	M	10	0	FACT.DBF
THRESHOLD	N	6	2	AREA.DBF
TOPIC	C	80	0	HELP.DBF
TOPIC	C	30	0	KBHELP.DBF
TREATMENT	M	10	0	DISEASE.DBF
UNITS	C	10	0	VAL.DBF
VAL	C	10	0	ACTION.DBF
VAL	C	10	0	FACT.DBF
WIDTH	N	2	0	VAL.DBF

Section V. Tree Diagram. The tree diagram shows the correlate among the function and procedure. The diagram lists each program in The order in which it is used. Each program has a list of all functions called and where the procedure/function is stored.

```
KBINIT.PRG
├────SETPATH()  (function in KBINIT.PRG)
├────KBMENU.MPR
│    ├────_QUIT()  (function in KBINIT.PRG)
│    │    └────ERRMSG.PRG
│    ├────BACKUP.PRG
│    │    └────BACKUP.SPR
│    │         ├────NEWDRIVEPOP   (procedure in BACKUP.SPR)
│    │         ├────NEWPATHPOP    (procedure in BACKUP.SPR)
│    │         ├────NEWFILEPOP    (procedure in BACKUP.SPR)
│    │         ├────_QSF0K096C()  (function in BACKUP.SPR)
│    │         ├────_QSF0K098Q()  (function in BACKUP.SPR)
│    │         │    ├────YESNO.PRG
│    │         │    ├────NEWPATHPOP...     (procedure in BACKUP.SPR)
│    │         │    └────NEWFILEPOP...     (procedure in BACKUP.SPR)
│    │         ├────_QSF0K09EC()  (function in BACKUP.SPR)
│    │         │    ├────PATHSTRING        (procedure in BACKUP.SPR)
│    │         │    ├────NEWPATHPOP...     (procedure in BACKUP.SPR)
│    │         │    └────NEWFILEPOP...     (procedure in BACKUP.SPR)
│    │         ├────_QSF0K09HK()  (function in BACKUP.SPR)
│    │         │    ├────YESNO.PRG...
│    │         │    └────PKZIP     (procedure in BACKUP.SPR)
│    │         │         └────PATHSTRING...     (procedure in BACKUP.SPR)
│    │         ├────_QSF0K09LV()  (function in BACKUP.SPR)
│    │         │    ├────NEWPATHPOP...     (procedure in BACKUP.SPR)
│    │         │    └────NEWFILEPOP...     (procedure in BACKUP.SPR)
│    │         ├────_QSF0KOAG9()  (function in BACKUP.SPR)
│    │         ├────_QSF0KOAID()  (function in BACKUP.SPR)
│    │         │    ├────YESNO.PRG...
│    │         │    ├────NEWPATHPOP...     (procedure in BACKUP.SPR)
│    │         │    └────NEWFILEPOP...     (procedure in BACKUP.SPR)
│    │         ├────_QSF0KOANU()  (function in BACKUP.SPR)
│    │         │    ├────PATHSTRING...     (procedure in BACKUP.SPR)
│    │         │    ├────NEWPATHPOP...     (procedure in BACKUP.SPR)
│    │         │    └────NEWFILEPOP...     (procedure in BACKUP.SPR)
│    │         ├────_QSF0KOAQV()  (function in BACKUP.SPR)
│    │         │    ├────YESNO.PRG...
│    │         │    └────PKZIP...     (procedure in BACKUP.SPR)
│    │         └────_QSF0KOAUY()  (function in BACKUP.SPR)
│    │              ├────NEWPATHPOP...     (procedure in BACKUP.SPR)
│    │              └────NEWFILEPOP...     (procedure in BACKUP.SPR)
│    ├────RESTORE.PRG
│    │    └────RESTORE.SPR
│    │         ├────NEWDRIVEPOP...    (procedure in BACKUP.SPR)
│    │         ├────NEWPATHPOP...     (procedure in BACKUP.SPR)
│    │         ├────NEWFILEPOP...     (procedure in BACKUP.SPR)
│    │         ├────_QSF0KPMPZ()  (function in RESTORE.SPR)
│    │         ├────_QSF0KPMS0()  (function in RESTORE.SPR)
│    │         │    ├────YESNO.PRG...
│    │         │    ├────NEWPATHPOP...     (procedure in BACKUP.SPR)
│    │         │    └────NEWFILEPOP...     (procedure in BACKUP.SPR)
│    │         │
│    │         ├────_QSF0KPMXI()  (function in RESTORE.SPR)
│    │         │    ├────PATHSTRING...      (procedure in BACKUP.SPR)
│    │         │    ├────NEWPATHPOP...      (procedure in BACKUP.SPR)
│    │         │    └────NEWFILEPOP...      (procedure in BACKUP.SPR)
│    │         ├────_QSF0KPN0J()  (function in RESTORE.SPR)
│    │         │    └────PKUNZIP     (procedure in RESTORE.SPR)
│    │         │         └────PATHSTRING...      (procedure in BACKUP.SPR)
│    │         ├────_QSF0KPN3T()  (function in RESTORE.SPR)
│    │         │    ├────NEWPATHPOP...      (procedure in BACKUP.SPR)
│    │         │    └────NEWFILEPOP...      (procedure in BACKUP.SPR)
│    │         ├────_QSF0KPN7E()  (function in RESTORE.SPR)
│    │         ├────_QSF0KPNY5()  (function in RESTORE.SPR)
│    │         ├────_QSF0KPO0A()  (function in RESTORE.SPR)
│    │         │    ├────YESNO.PRG...
│    │         │    ├────NEWPATHPOP...      (procedure in BACKUP.SPR)
│    │         │    └────NEWFILEPOP...      (procedure in BACKUP.SPR)
│    │         ├────_QSF0KPO5T()  (function in RESTORE.SPR)
│    │         │    ├────PATHSTRING...      (procedure in BACKUP.SPR)
│    │         │    ├────NEWPATHPOP...      (procedure in BACKUP.SPR)
│    │         │    └────NEWFILEPOP...      (procedure in BACKUP.SPR)
│    │         ├────_QSF0KPO8T()  (function in RESTORE.SPR)
│    │         │    └────PKUNZIP...     (procedure in RESTORE.SPR)
│    │         ├────_QSF0KPOBY()  (function in RESTORE.SPR)
│    │         │    ├────NEWPATHPOP...      (procedure in BACKUP.SPR)
│    │         │    └────NEWFILEPOP...      (procedure in BACKUP.SPR)
│    │         └────_QSF0KPOFG()  (function in RESTORE.SPR)
│    ├────KB.SPR
│    │    ├────SETQUALS      (procedure in KB.SPR)
│    │    ├────SETGOALS      (procedure in KB.SPR)
│    │    ├────SETDISPLAY    (procedure in KB.SPR)
│    │    ├────_QSF0KOFX9()  (function in KB.SPR)
│    │    │    ├────KBLOAD.SPR
│    │    │    │    ├────_QSF0KQBNK()  (function in KBLOAD.SPR)
│    │    │    │    │    └────LOADOK     (procedure in KBLOAD.SPR)
│    │    │    │    ├────_QSF0KQBT0()  (function in KBLOAD.SPR)
│    │    │    │    │    └────LOADOK...     (procedure in KBLOAD.SPR)
│    │    │    │    ├────_QSF0KQC1J()  (function in KBLOAD.SPR)
│    │    │    │    │    └────LOADOK...     (procedure in KBLOAD.SPR)
│    │    │    │    ├────_QSF0KQC42()  (function in KBLOAD.SPR)
│    │    │    │    │    └────LOADOK...     (procedure in KBLOAD.SPR)
│    │    │    │    ├────LOADOK...     (procedure in KBLOAD.SPR)
│    │    │    │    ├────_QSF0KQC6T()  (function in KBLOAD.SPR)
│    │    │    │    │    ├────CHECKFILE()  (function in KBLOAD.SPR)
│    │    │    │    │    │    └────YESNO.PRG...
│    │    │    │    │    ├────POPUPSHOW()  (function in KBINIT.PRG)
│    │    │    │    │    ├────KBLDR.PRG
│    │    │    │    │    └────POPUPHIDE()  (function in KBINIT.PRG)
│    │    │    │    ├────_QSF0KQCP2()  (function in KBLOAD.SPR)
│    │    │    │    │    └────LOADOK...     (procedure in KBLOAD.SPR)
│    │    │    │    ├────_QSF0KQCRO()  (function in KBLOAD.SPR)
│    │    │    │    │    └────LOADOK...     (procedure in KBLOAD.SPR)
│    │    │    │    ├────_QSF0KQCUA()  (function in KBLOAD.SPR)
│    │    │    │    │    └────LOADOK...     (procedure in KBLOAD.SPR)
│    │    │    │    ├────_QSF0KQCWT()  (function in KBLOAD.SPR)
│    │    │    │    │    └────LOADOK...     (procedure in KBLOAD.SPR)
│    │    │    │    └────_QSF0KQCZG()  (function in KBLOAD.SPR)
│    │    │    │         └────KBLDR.PRG...
│    │    │    │
│    │    │    ├────KBEDIT.SPR
│    │    │    │    ├────SETPREM       (procedure in KBEDIT.SPR)
│    │    │    │    │    ├────SEMPTY() (function in KBEDIT.SPR)
│    │    │    │    │    ├────HLINK()  (function in KBEDIT.SPR)
│    │    │    │    │    └────VLINK()  (function in KBEDIT.SPR)
│    │    │    │    ├────SETACT        (procedure in KBEDIT.SPR)
│    │    │    │    ├────SETELSE       (procedure in KBEDIT.SPR)
│    │    │    │    ├────SETACTION()   (function in KBEDIT.SPR)
│    │    │    │    ├────_QSF0KQ0SA()  (function in KBEDIT.SPR)
│    │    │    │    │    ├────TERM.SPR
│    │    │    │    │    │    ├────ADDFACT()  (function in TERM.SPR)
│    │    │    │    │    │    │    ├────CLAUSE.SPR
│    │    │    │    │    │    │    │    ├────SETOBJECT()   (function in
│    │    │    │    │    │    │    │    │                   CLAUSE.SPR)
│    │    │    │    │    │    │    │    ├────_QSF0KP3SZ()  (function in
│    │    │    │    │    │    │    │    │                   CLAUSE.SPR)
│    │    │    │    │    │    │    │    ├────_QSF0KP3X2()  (function in
│    │    │    │    │    │    │    │    │                   CLAUSE.SPR)
│    │    │    │    │    │    │    │    ├────_QSF0KP435()  (function in
│    │    │    │    │    │    │    │    │                   CLAUSE.SPR)
│    │    │    │    │    │    │    │    ├────_QSF0KP472()  (function in
│    │    │    │    │    │    │    │    │                   CLAUSE.SPR)
│    │    │    │    │    │    │    │    ├────INSNEWID() (function CLAUSE.SPR)
│    │    │    │    │    │    │    │    │    └────DP.PRG
│    │    │    │    │    │    │    │    ├────_QSF0KP5HS()  (function in
│    │    │    │    │    │    │    │    │                   CLAUSE.SPR)
│    │    │    │    │    │    │    │    ├────_QSF0KP5LZ()  (function in
│    │    │    │    │    │    │    │    │                   CLAUSE.SPR)
│    │    │    │    │    │    │    │    ├────_QSF0KP5Q1()  (function in
│    │    │    │    │    │    │    │    │                   CLAUSE.SPR)
│    │    │    │    │    │    │    │    └────_QSF0KP5UC()  (function in
│    │    │    │    │    │    │    │                        CLAUSE.SPR)
│    │    │    │    │    │    │    └────SETPREM...(procedure in KBEDIT.SPR)
│    │    │    │    │    │    ├────_QSF0KPC3R()  (function in TERM.SPR)
│    │    │    │    │    │    │    └────EDPREM     (procedure in TERM.SPR)
│    │    │    │    │    │    │         ├────CLAUSE.SPR...
│    │    │    │    │    │    │         └────SETPREM...     (procedure in
│    │    │    │    │    │    │                              KBEDIT.SPR)
│    │    │    │    │    │    ├────_QSF0KPC6D()  (function in TERM.SPR)
│    │    │    │    │    │    │    └────EDPREM...      (procedure in TERM.SPR)
│    │    │    │    │    │    ├────_QSF0KPCPM()  (function in TERM.SPR)
│    │    │    │    │    │    ├────_QSF0KPCU9()  (function in TERM.SPR)
│    │    │    │    │    │    │    └────SETJOIN()  (function in TERM.SPR)
│    │    │    │    │    │    │         └────SETPREM...     (procedure in
│    │    │    │    │    │    │                              KBEDIT.SPR)
│    │    │    │    │    │    ├────_QSF0KPCWU()  (function in TERM.SPR)
│    │    │    │    │    │    │    └────SETJOIN() ... (function in TERM.SPR)
│    │    │    │    │    │    ├────_QSF0KPCZG()  (function in TERM.SPR)
│    │    │    │    │    │    │    └────EDPREM...      (procedure in TERM.SPR)
│    │    │    │    │    │    ├────_QSF0KPD1Z()  (function in TERM.SPR)
│    │    │    │    │    │    │    └────ADDFACT() ... (function in TERM.SPR).
│    │    │    │    │    │    └────_QSF0KPD51()  (function in TERM.SPR)
│    │    │    │    │    │         └────EDPREM...      (procedure in TERM.SPR)
│    │    │    │    │    └────SETPREM...     (procedure in KBEDIT.SPR)
│    │    │    │    ├────_QSF0KQ0WK()  (function in KBEDIT.SPR)
│    │    │    │    │    └────ACTION.SPR
│    │    │    │    │         ├────ERRMSG.PRG...
│   │   │   │   │       ├───_QSF0KOX5V()  (function in ACTION.SPR)
│   │   │   │   │       │   └───EDACT     (procedure in ACTION.SPR)
│   │   │   │   │       │       ├───CLAUSE.SPR...
│   │   │   │   │       │       └───SETACT...     (procedure in
│   │   │   │   │       │                          KBEDIT.SPR)
│   │   │   │   │       ├───_QSF0KOX9D()  (function in ACTION.SPR)
│   │   │   │   │       │   └───EDACT...     (procedure in ACTION.SPR)
│   │   │   │   │       ├───ADDNEWACT()  (function in ACTION.SPR)
│   │   │   │   │       │   ├───CLAUSE.SPR...
│   │   │   │   │       │   └───SETACT... (procedure in KBEDIT.SPR)
│   │   │   │   │       ├───_QSF0KOXVA()  (function in ACTION.SPR)
│   │   │   │   │       ├───_QSF0KOY5O()  (function in ACTION.SPR)
│   │   │   │   │       │   └───EDACT...     (procedure in ACTION.SPR)
│   │   │   │   │       ├───_QSF0KOY8A()  (function in ACTION.SPR)
│   │   │   │   │       │   ├───CLAUSE.SPR...
│   │   │   │   │       │   └───SETACT... (procedure in KBEDIT.SPR)
│   │   │   │   │       └───_QSF0KOYCP()  (function in ACTION.SPR)
│   │   │   │   │           └───EDACT...     (procedure in ACTION.SPR)
│   │   │   │   ├───_QSF0KQ10R()  (function in KBEDIT.SPR)
│   │   │   │   │   ├───ACTELSE.SPR
│   │   │   │   │   │   ├───ERRMSG.PRG...
│   │   │   │   │   │   ├───_QSF0KP07D()  (function in ACTELSE.SPR)
│   │   │   │   │   │   │   └───EDELSE     (procedure in ACTELSE.SPR)
│   │   │   │   │   │   │       ├───CLAUSE.SPR...
│   │   │   │   │   │   │       └───SETELSE...     (procedure in
│   │   │   │   │   │   │                           KBEDIT.SPR)
│   │   │   │   │   │   ├───_QSF0KP0AR()  (function in ACTELSE.SPR)
│   │   │   │   │   │   │   └───EDELSE... (procedure in ACTELSE.SPR)
│   │   │   │   │   │   ├───ADDNEWELSE()  (function in ACTELSE.SPR)
│   │   │   │   │   │   │   ├───CLAUSE.SPR...
│   │   │   │   │   │   │   └───SETELSE...(procedure in KBEDIT.SPR)
│   │   │   │   │   │   ├───_QSF0KP0S3()  (function in ACTELSE.SPR)
│   │   │   │   │   │   ├───_QSF0KP0W3()  (function in ACTELSE.SPR)
│   │   │   │   │   │   │   └───EDELSE... (procedure in ACTELSE.SPR)
│   │   │   │   │   │   ├───_QSF0KP0YP()  (function in ACTELSE.SPR)
│   │   │   │   │   │   │   ├───ADDNEWELSE() ... (function in
│   │   │   │   │   │   │   │                     ACTELSE.SPR)
│   │   │   │   │   │   │   ├───CLAUSE.SPR...
│   │   │   │   │   │   │   └───SETELSE...(procedure in KBEDIT.SPR)
│   │   │   │   │   │   └───_QSF0KP13M()  (function in ACTELSE.SPR)
│   │   │   │   │   │       └───EDELSE... (procedure in ACTELSE.SPR)
│   │   │   │   │   └───SETELSE...     (procedure in KBEDIT.SPR)
│   │   │   │   └───_QSF0KQ14Z()  (function in KBEDIT.SPR)
│   │   │   │       ├───RULE.SPR
│   │   │   │       │   ├───_QSF0KP97Y()  (function in RULE.SPR)
│   │   │   │       │   ├───_QSF0KPA3B()  (function in RULE.SPR)
│   │   │   │       │   └───_QSF0KPA7J()  (function in RULE.SPR)
│   │   │   │       │       └───YESNO.PRG...
│   │   │   │       ├───SETPREM...     (procedure in KBEDIT.SPR)
│   │   │   │       ├───SETACT...     (procedure in KBEDIT.SPR)
│   │   │   │       ├───SETELSE...     (procedure in KBEDIT.SPR)
│   │   │   │       └───SETACTION() ... (function in KBEDIT.SPR)
│   │   │   ├───KBDEL.SPR
│   │   │   │   ├───_QSF0KOKEU()  (function in KBDEL.SPR)
│   │   │   │   │   └───KBDELETE.PRG
│   │   │   │   └───_QSF0KOKPT()  (function in KBDEL.SPR)
│   │   │   │       └───KBDELETE.PRG...
│   │   │   │
│    │    │    ├────SETQUALS...     (procedure in KB.SPR)
│    │    │    ├────SETGOALS...     (procedure in KB.SPR)
│    │    │    └────SETDISPLAY...     (procedure in KB.SPR)
│    │    ├────_QSF0KOG2M()  (function in KB.SPR)
│    │    │    ├────QUAL.SPR
│    │    │    │    ├────_QSF0KOMJF()  (function in QUAL.SPR)
│    │    │    │    │    └────EDQUAL     (procedure in KB.SPR)
│    │    │    │    │         └────EDOBJ()  (function in KB.SPR)
│    │    │    │    │              ├────DISEASE.SPR
│    │    │    │    │              │    ├────_QSF0KOU0D()  (function in
│    │    │    │    │              │    │                   DISEASE.SPR)
│    │    │    │    │              │    ├────_QSF0KOU3P()  (function in
│    │    │    │    │              │    │                   DISEASE.SPR)
│    │    │    │    │              │    ├────_QSF0KOUMI()  (function in
│    │    │    │    │              │    │                   DISEASE.SPR)
│    │    │    │    │              │    └────_QSF0KOUQK()  (function in
│    │    │    │    │              │                        DISEASE.SPR)
│    │    │    │    │              └────DICT.SPR
│    │    │    │    │                   ├────SETENUM (procedure in DICT.SPR)
│    │    │    │    │                   ├────_QSF0KPY4X() (funct in DICT.SPR)
│    │    │    │    │                   ├────_QSF0KPY8W() (funct in DICT.SPR)
│    │    │    │    │                   │    └────EDENUM     (procedure in
│    │    │    │    │                   │         │           DICT.SPR)
│    │    │    │    │                   │         ├────ENUM.SPR
│    │    │    │    │                   │         │    ├────_QSF0KPJAM()
│    │    │    │    │                   │         │    │     (function in
│    │    │    │    │                   │         │    │      ENUM.SPR)
│    │    │    │    │                   │         │    ├────_QSF0KPJF7()
│    │    │    │    │                   │         │    │(function in ENUM.SPR)
│    │    │    │    │                   │         │    ├────_QSF0KPJJK()
│    │    │    │    │                   │         │    │(function in ENUM.SPR)
│    │    │    │    │                   │         │    ├────_QSF0KPJYY()
│    │    │    │    │                   │         │    │(function in ENUM.SPR)
│    │    │    │    │                   │         │    ├────_QSF0KPK3S()
│    │    │    │    │                   │         │    │(function in ENUM.SPR)
│    │    │    │    │                   │         │    └────_QSF0KPK8E()
│    │    │    │    │                   │         │     (function in ENUM.SPR)
│    │    │    │    │                   │         ├────SETENUM. (procedure
│    │    │    │    │                   │         │              in DICT.SPR)
│    │    │    │    │                   │         └────DDENUM.SPR
│    │    │    │    │                   │              ├────_QSF0KQ9MB()
│    │    │    │    │                   │              │(funct in DDENUM.SPR)
│    │    │    │    │                   │              └────_QSF0KQ9QC()
│    │    │    │    │                   │               (funct in DDENUM.SPR)
│    │    │    │    │                   ├────_QSF0KPYF6()  (funct in DICT.SPR)
│    │    │    │    │                   └────_QSF0KPYJ0()  (funct in DICT.SPR)
│    │    │    │    ├────_QSF0KOMRX()  (function in QUAL.SPR)
│    │    │    │    │    └────EDQUAL...     (procedure in KB.SPR)
│    │    │    │    ├────_QSF0KON7E()  (function in QUAL.SPR)
│    │    │    │    │    ├────OBJECT.SPR
│    │    │    │    │    │    ├────_QSF0KPFR1()  (function in OBJECT.SPR)
│    │    │    │    │    │    ├────_QSF0KPFV8()  (function in OBJECT.SPR)
│    │    │    │    │    │    │    └────GETOBJ()  (function in OBJECT.SPR)
│    │    │    │    │    │    │         ├────CREATARRAY()  (function in
│    │    │    │    │    │    │         │                   OBJECT.SPR)
│    │    │    │    │    │    │         └────SELITEM() (funct in OBJECT.SPR)
│    │    │    │    │    │    │              ├────OBLIST.SPR
│    │    │    │    │    │    │              │    └────_QSF0KQ4M3() (funct in
│    │    │    │    │    │    │              │                   OBLIST.SPR)
│    │    │    │    │    │    │              ├────DISEASE.SPR...
│    │    │    │    │    │    │              └────DICT.SPR...
│    │    │    │    │    │    ├────_QSF0KPFYY()  (function in OBJECT.SPR)
│    │    │    │    │    │    ├────_QSF0KPG47()  (function in OBJECT.SPR)
│    │    │    │    │    │    │    ├────CREATARRAY() ... (funct in OBJECT.SPR)
│    │    │    │    │    │    │    └────SELITEM() ... (funct in OBJECT.SPR)
│    │    │    │    │    │    ├────_QSF0KPGOJ()  (function in OBJECT.SPR)
│    │    │    │    │    │    ├────_QSF0KPGSR()  (function in OBJECT.SPR)
│    │    │    │    │    │    │    └────GETOBJ() ... (function in OBJECT.SPR)
│    │    │    │    │    │    ├────_QSF0KPGVZ()  (function in OBJECT.SPR)
│    │    │    │    │    │    └────_QSF0KPH17()  (function in OBJECT.SPR)
│    │    │    │    │    │         ├────CREATARRAY() ... (funct in OBJECT.SPR)
│    │    │    │    │    │         └────SELITEM() ... (funct in OBJECT.SPR)
│    │    │    │    │    ├────SETQUALS...      (procedure in KB.SPR)
│    │    │    │    │    ├────VALIDOBJ()  (function in QUAL.SPR)
│    │    │    │    │    │    └────ERRMSG.PRG...
│    │    │    │    │    └────QUALDEL()  (function in QUAL.SPR)
│    │    │    │    │         ├────POPUPSHOW() ... (function in KBINIT.PRG)
│    │    │    │    │         ├────POPUPHIDE() ... (function in KBINIT.PRG)
│    │    │    │    │         └────SETQUALS...     (procedure in KB.SPR)
│    │    │    │    └────_QSF0KONBU()  (function in QUAL.SPR)
│    │    │    │         └────EDQUAL...     (procedure in KB.SPR)
│    │    │    ├────GOAL.SPR
│    │    │    │    ├────_QSF0KOPCM()  (function in GOAL.SPR)
│    │    │    │    │    └────EDGOAL     (procedure in KB.SPR)
│    │    │    │    │         └────EDOBJ() ... (function in KB.SPR)
│    │    │    │    ├────_QSF0KOPG7()  (function in GOAL.SPR)
│    │    │    │    │    └────EDGOAL...     (procedure in KB.SPR)
│    │    │    │    ├────_QSF0KOPTK()  (function in GOAL.SPR)
│    │    │    │    │    ├────OBJECT.SPR...
│    │    │    │    │    └────SETGOALS...      (procedure in KB.SPR)
│    │    │    │    └────_QSF0KOPXH()  (function in GOAL.SPR)
│    │    │    │         └────EDGOAL...     (procedure in KB.SPR)
│    │    │    ├────DISPLAY.SPR
│    │    │    │    ├────_QSF0KORJ1()  (function in DISPLAY.SPR)
│    │    │    │    │    └────EDDISPLAY     (procedure in KB.SPR)
│    │    │    │    │         └────EDOBJ() ... (function in KB.SPR)
│    │    │    │    ├────_QSF0KORMJ()  (function in DISPLAY.SPR)
│    │    │    │    │    └────EDDISPLAY...     (procedure in KB.SPR)
│    │    │    │    ├────_QSF0KORZX()  (function in DISPLAY.SPR)
│    │    │    │    ├────_QSF0KOS30()  (function in DISPLAY.SPR)
│    │    │    │    └────_QSF0KOS6A()  (function in DISPLAY.SPR)
│    │    │    │         └────EDDISPLAY...     (procedure in KB.SPR)
│    │    │    ├────SETQUALS...     (procedure in KB.SPR)
│    │    │    ├────SETGOALS...     (procedure in KB.SPR)
│    │    │    └────SETDISPLAY...     (procedure in KB.SPR)
│    │    ├────_QSF0KOGFV()  (function in KB.SPR)
│    │    ├────_QSF0KOGIH()  (function in KB.SPR)
│    │    ├────_QSF0KOH4Z()  (function in KB.SPR)
│    │    │    ├────KBLOAD.SPR...
│    │    │    ├────KBDEL.SPR...
│    │    │    ├────DISPLAY.SPR...
│    │    │    ├────KB.SPR...
│    │    │    ├────RULE.SPR...
│    │    │    ├────QUAL.SPR...
│    │    │    ├────GOAL.SPR...
│    │    │    ├────SETPREM...     (procedure in KBEDIT.SPR)
│    │    │    ├────SETACT...     (procedure in KBEDIT.SPR)
│    │    │    ├────SETQUALS...     (procedure in KB.SPR)
│    │    │    ├────SETGOALS...     (procedure in KB.SPR)
│    │    │    └────SETDISPLAY...     (procedure in KB.SPR)
│    │    ├────_QSF0KOHJ1()  (function in KB.SPR)
│    │    ├────_QSF0KOHM5()  (function in KB.SPR)
│    │    └────_QSF0KOHPT()  (function in KB.SPR)
│    └────DD.SPR
│         ├────_QSF0KPT2I()  (function in DD.SPR)
│         │    ├────DDEDIT.SPR
│         │    │    ├────SETRULES()  (function in DDEDIT.SPR)
│         │    │    │    └────AREANAME()  (function in DDEDIT.SPR)
│         │    │    ├────SETENUM...     (procedure in DICT.SPR)
│         │    │    ├────_QSF0KQ6M1()  (function in DDEDIT.SPR)
│         │    │    │    ├────SETVAL      (procedure in DICT.SPR)
│         │    │    │    ├────SETENUM...     (procedure in DICT.SPR)
│         │    │    │    └────DDENUM.SPR...
│         │    │    ├────_QSF0KQ6T0()  (function in DDEDIT.SPR)
│         │    │    │    └────EDENUM...     (procedure in DICT.SPR)
│         │    │    ├────_QSF0KQ6YY()  (function in DDEDIT.SPR)
│         │    │    │    └────CLEANENUM()  (function in DDEDIT.SPR)
│         │    │    └────_QSF0KQ74F()  (function in DDEDIT.SPR)
│         │    ├────VALIDOBJ() ... (function in QUAL.SPR)
│         │    ├────QUALDEL() ... (function in QUAL.SPR)
│         │    └────RESETDATA()  (function in DD.SPR)
│         └────_QSF0KPT7D()  (function in DD.SPR)
│              └────DDEDIT.SPR...
└────MYHANDLER()  (function in KBINIT.PRG)
```

Section VI. Procedure and Function Summary. The system contains 24 procedure files: KBINIT.PRG, KBLOAD.SPR, KB.SPR, KBDEL.SPR, QUAL.SPR, GOAL.SPR, DISPLAY.SPR, DISEASE.SPR, ACTION.SPR, ACTELSE.SPR, CLAUSE.SPR, RULE.SPR, TERM.SPR, OBJECT.SPR, ENUM.SPR, RESTORE.SPR, DD.SPR, PLAN.SPR, DICT.SPR, KBEDIT.SPR, OBLIST.SPR, DDEDIT.SPR, DDENUM.SPR, and BACKUP.SPR

1. **KBINIT.PRG**

```
Contains: MYHANDLER()                (Params: none)
   Called by: KBINIT.PRG
Contains: _QUIT()                    (Params: none)
   Called by: KBMENU.MPR
       Calls: ERRMSG.PRG
Contains: POPUPSHOW()                (Params: ERRSTR)
   Called by: QUALDEL()              (function in QUAL.SPR)
   Called by: _QSF0KQC6T()           (function in KBLOAD.SPR)
Contains: POPUPHIDE()                (Params: W)
   Called by: QUALDEL()              (function in QUAL.SPR)
   Called by: _QSF0KQC6T()           (function in KBLOAD.SPR)
Contains: SETPATH()                  (Params: NEWPATH)
   Called by: KBINIT.PRG
```

2. **KBLOAD.SPR**

```
Contains: _QSF0KQBNK()               (Params: none)
   Called by: KBLOAD.SPR
       Calls: LOADOK                 (procedure in KBLOAD.SPR)
Contains: _QSF0KQBT0()               (Params: none)
   Called by: KBLOAD.SPR
       Calls: LOADOK                 (procedure in KBLOAD.SPR)
Contains: _QSF0KQC1J()               (Params: none)
   Called by: KBLOAD.SPR
       Calls: LOADOK                 (procedure in KBLOAD.SPR)
Contains: _QSF0KQC42()               (Params: none)
   Called by: KBLOAD.SPR
       Calls: LOADOK                 (procedure in KBLOAD.SPR)
Contains: _QSF0KQC6T()               (Params: none)
   Called by: KBLOAD.SPR
       Calls: CHECKFILE()            (function in KBLOAD.SPR)
       Calls: POPUPSHOW()            (function in KBINIT.PRG)
       Calls: KBLDR.PRG
       Calls: POPUPHIDE()            (function in KBINIT.PRG)
Contains: _QSF0KQCP2()               (Params: none)
   Called by: KBLOAD.SPR
       Calls: LOADOK                 (procedure in KBLOAD.SPR)
Contains: _QSF0KQCRO()               (Params: none)
   Called by: KBLOAD.SPR
       Calls: LOADOK                 (procedure in KBLOAD.SPR)
Contains: _QSF0KQCUA()               (Params: none)
   Called by: KBLOAD.SPR
       Calls: LOADOK                 (procedure in KBLOAD.SPR)
Contains: _QSF0KQCWT()               (Params: none)
   Called by: KBLOAD.SPR
       Calls: LOADOK                 (procedure in KBLOAD.SPR)
Contains: _QSF0KQCZG()               (Params: none)
```

```
      Called by: KBLOAD.SPR
          Calls: KBLDR.PRG
   Contains: LOADOK                      (Params: none)
      Called by: KBLOAD.SPR
      Called by: _QSF0KQBNK()            (function in KBLOAD.SPR)
      Called by: _QSF0KQBT0()            (function in KBLOAD.SPR)
      Called by: _QSF0KQC1J()            (function in KBLOAD.SPR)
      Called by: _QSF0KQC42()            (function in KBLOAD.SPR)
      Called by: _QSF0KQCP2()            (function in KBLOAD.SPR)
      Called by: _QSF0KQCRO()            (function in KBLOAD.SPR)
      Called by: _QSF0KQCUA()            (function in KBLOAD.SPR)
      Called by: _QSF0KQCWT()            (function in KBLOAD.SPR)
   Contains: CHECKFILE()                 (Params: M.NEW)
      Called by: _QSF0KQC6T()            (function in KBLOAD.SPR)
          Calls: YESNO.PRG
```

3. KB.SPR

```
   Contains: _QSF0KOFX9()                (Params: none)
      Called by: KB.SPR
          Calls: KBLOAD.SPR
          Calls: KBEDIT.SPR
          Calls: KBDEL.SPR
          Calls: SETQUALS                (procedure in KB.SPR)
          Calls: SETGOALS                (procedure in KB.SPR)
          Calls: SETDISPLAY              (procedure in KB.SPR)
   Contains: _QSF0KOG2M()                (Params: none)
      Called by: KB.SPR
          Calls: QUAL.SPR
          Calls: GOAL.SPR
          Calls: DISPLAY.SPR
          Calls: SETQUALS                (procedure in KB.SPR)
          Calls: SETGOALS                (procedure in KB.SPR)
          Calls: SETDISPLAY              (procedure in KB.SPR)
   Contains: _QSF0KOGFV()                (Params: none)
      Called by: KB.SPR
   Contains: _QSF0KOGIH()                (Params: none)
      Called by: KB.SPR
   Contains: _QSF0KOH4Z()                (Params: none)
      Called by: KB.SPR
          Calls: KBLOAD.SPR
          Calls: KBDEL.SPR
          Calls: DISPLAY.SPR
          Calls: KB.SPR
          Calls: RULE.SPR
          Calls: QUAL.SPR
          Calls: GOAL.SPR
          Calls: SETPREM                 (procedure in KBEDIT.SPR)
          Calls: SETACT                  (procedure in KBEDIT.SPR)
          Calls: SETQUALS                (procedure in KB.SPR)
          Calls: SETGOALS                (procedure in KB.SPR)
          Calls: SETDISPLAY              (procedure in KB.SPR)
   Contains: _QSF0KOHJ1()                (Params: none)
      Called by: KB.SPR
   Contains: _QSF0KOHM5()                (Params: none)
      Called by: KB.SPR
   Contains: _QSF0KOHPT()                (Params: none)
```

```
      Called by: KB.SPR
   Contains: SETQUALS                 (Params: none)
      Called by: KB.SPR
      Called by: _QSF0KOFX9()         (function in KB.SPR)
      Called by: _QSF0KOG2M()         (function in KB.SPR)
      Called by: _QSF0KOH4Z()         (function in KB.SPR)
      Called by: _QSF0KON7E()         (function in QUAL.SPR)
      Called by: QUALDEL()            (function in QUAL.SPR)
   Contains: SETGOALS                 (Params: none)
      Called by: KB.SPR
      Called by: _QSF0KOFX9()         (function in KB.SPR)
      Called by: _QSF0KOG2M()         (function in KB.SPR)
      Called by: _QSF0KOH4Z()         (function in KB.SPR)
      Called by: _QSF0KOPTK()         (function in GOAL.SPR)
      Called by: _QSF0KPV8R()         (function in PLAN.SPR)
      Called by: _QSF0KPVQB()         (function in PLAN.SPR)
   Contains: SETDISPLAY               (Params: none)
      Called by: KB.SPR
      Called by: _QSF0KOFX9()         (function in KB.SPR)
      Called by: _QSF0KOG2M()         (function in KB.SPR)
      Called by: _QSF0KOH4Z()         (function in KB.SPR)
   Contains: EDGOAL                   (Params: none)
      Called by: _QSF0KOPCM()         (function in GOAL.SPR)
      Called by: _QSF0KOPG7()         (function in GOAL.SPR)
      Called by: _QSF0KOPXH()         (function in GOAL.SPR)
      Called by: _QSF0KPVCR()         (function in PLAN.SPR)
      Called by: _QSF0KPVW8()         (function in PLAN.SPR)
          Calls: EDOBJ()              (function in KB.SPR)
   Contains: EDQUAL                   (Params: none)
      Called by: _QSF0KOMJF()         (function in QUAL.SPR)
      Called by: _QSF0KOMRX()         (function in QUAL.SPR)
      Called by: _QSF0KONBU()         (function in QUAL.SPR)
          Calls: EDOBJ()              (function in KB.SPR)
   Contains: EDDISPLAY                (Params: none)
      Called by: _QSF0KORJ1()         (function in DISPLAY.SPR)
      Called by: _QSF0KORMJ()         (function in DISPLAY.SPR)
      Called by: _QSF0KOS6A()         (function in DISPLAY.SPR)
          Calls: EDOBJ()              (function in KB.SPR)
   Contains: EDOBJ()                  (Params: MOBJ, MID)
      Called by: EDGOAL               (procedure in KB.SPR)
      Called by: EDQUAL               (procedure in KB.SPR)
      Called by: EDDISPLAY            (procedure in KB.SPR)
          Calls: DISEASE.SPR
          Calls: DICT.SPR
```

4. KBDEL.SPR

```
   Contains: _QSF0KOKEU()             (Params: none)
      Called by: KBDEL.SPR
          Calls: KBDELETE.PRG
   Contains: _QSF0KOKPT()             (Params: none)
      Called by: KBDEL.SPR
          Calls: KBDELETE.PRG
```

5. QUAL.SPR

```
   Contains: _QSF0KOMJF()             (Params: none)
```

```
    Called by: QUAL.SPR
        Calls: EDQUAL                    (procedure in KB.SPR)
Contains: _QSF0KOMRX()                   (Params: none)
    Called by: QUAL.SPR
        Calls: EDQUAL                    (procedure in KB.SPR)
Contains: _QSF0KON7E()                   (Params: none)
    Called by: QUAL.SPR
        Calls: OBJECT.SPR
        Calls: SETQUALS                  (procedure in KB.SPR)
        Calls: VALIDOBJ()                (function in QUAL.SPR)
        Calls: QUALDEL()                 (function in QUAL.SPR)
Contains: _QSF0KONBU()                   (Params: none)
    Called by: QUAL.SPR
        Calls: EDQUAL                    (procedure in KB.SPR)
Contains: POPUPSHOW()                    (Params: ERRSTR)
    Called by: QUALDEL()                 (function in QUAL.SPR)
    Called by: _QSF0KQC6T()              (function in KBLOAD.SPR)
Contains: POPUPHIDE()                    (Params: W)
    Called by: QUALDEL()                 (function in QUAL.SPR)
    Called by: _QSF0KQC6T()              (function in KBLOAD.SPR)
Contains: VALIDOBJ()                     (Params: none)
    Called by: _QSF0KON7E()              (function in QUAL.SPR)
    Called by: _QSF0KPT2I()              (function in DD.SPR)
        Calls: ERRMSG.PRG
Contains: QUALDEL()                      (Params: none)
    Called by: _QSF0KON7E()              (function in QUAL.SPR)
    Called by: _QSF0KPT2I()              (function in DD.SPR)
        Calls: POPUPSHOW()               (function in KBINIT.PRG)
        Calls: POPUPHIDE()               (function in KBINIT.PRG)
        Calls: SETQUALS                  (procedure in KB.SPR)
```

6. GOAL.SPR

```
Contains: _QSF0KOPCM()                   (Params: none)
    Called by: GOAL.SPR
        Calls: EDGOAL                    (procedure in KB.SPR)
Contains: _QSF0KOPG7()                   (Params: none)
    Called by: GOAL.SPR
        Calls: EDGOAL                    (procedure in KB.SPR)
Contains: _QSF0KOPTK()                   (Params: none)
    Called by: GOAL.SPR
        Calls: OBJECT.SPR
        Calls: SETGOALS                  (procedure in KB.SPR)
Contains: _QSF0KOPXH()                   (Params: none)
    Called by: GOAL.SPR
        Calls: EDGOAL                    (procedure in KB.SPR)
```

7. DISPLAY.SPR

```
Contains: _QSF0KORJ1()                   (Params: none)
    Called by: DISPLAY.SPR
        Calls: EDDISPLAY                 (procedure in KB.SPR)
Contains: _QSF0KORMJ()                   (Params: none)
    Called by: DISPLAY.SPR
        Calls: EDDISPLAY                 (procedure in KB.SPR)
Contains: _QSF0KORZX()                   (Params: none)
    Called by: DISPLAY.SPR
```

```
   Contains: _QSF0KOS30()                  (Params: none)
      Called by: DISPLAY.SPR
   Contains: _QSF0KOS6A()                  (Params: none)
      Called by: DISPLAY.SPR
          Calls: EDDISPLAY                 (procedure in KB.SPR)
```

8. DISEASE.SPR

```
   Contains: _QSF0KOU0D()                  (Params: none)
      Called by: DISEASE.SPR
   Contains: _QSF0KOU3P()                  (Params: none)
      Called by: DISEASE.SPR
   Contains: _QSF0KOUMI()                  (Params: none)
      Called by: DISEASE.SPR
   Contains: _QSF0KOUQK()                  (Params: none)
      Called by: DISEASE.SPR
```

9. ACTION.SPR

```
   Contains: _QSF0KOX5V()                  (Params: none)
      Called by: ACTION.SPR
          Calls: EDACT                     (procedure in ACTION.SPR)
   Contains: _QSF0KOX9D()                  (Params: none)
      Called by: ACTION.SPR
          Calls: EDACT                     (procedure in ACTION.SPR)
   Contains: _QSF0KOXVA()                  (Params: none)
      Called by: ACTION.SPR
   Contains: _QSF0KOY5O()                  (Params: none)
      Called by: ACTION.SPR
          Calls: EDACT                     (procedure in ACTION.SPR)
   Contains: _QSF0KOY8A()                  (Params: none)
      Called by: ACTION.SPR
          Calls: CLAUSE.SPR
          Calls: SETACT                    (procedure in KBEDIT.SPR)
   Contains: _QSF0KOYCP()                  (Params: none)
      Called by: ACTION.SPR
          Calls: EDACT                     (procedure in ACTION.SPR)
   Contains: EDACT                         (Params: none)
      Called by: _QSF0KOX5V()              (function in ACTION.SPR)
      Called by: _QSF0KOX9D()              (function in ACTION.SPR)
      Called by: _QSF0KOY5O()              (function in ACTION.SPR)
      Called by: _QSF0KOYCP()              (function in ACTION.SPR)
          Calls: CLAUSE.SPR
          Calls: SETACT                    (procedure in KBEDIT.SPR)
   Contains: ADDNEWACT()                   (Params: none)
      Called by: ACTION.SPR
          Calls: CLAUSE.SPR
          Calls: SETACT                    (procedure in KBEDIT.SPR)
```

10. ACTELSE.SPR

```
   Contains: _QSF0KP07D()                  (Params: none)
      Called by: ACTELSE.SPR
          Calls: EDELSE                    (procedure in ACTELSE.SPR)
   Contains: _QSF0KP0AR()                  (Params: none)
      Called by: ACTELSE.SPR
          Calls: EDELSE                    (procedure in ACTELSE.SPR)
```

```
Contains: _QSF0KP0S3()                 (Params: none)
   Called by: ACTELSE.SPR
Contains: _QSF0KP0W3()                 (Params: none)
   Called by: ACTELSE.SPR
       Calls: EDELSE                   (procedure in ACTELSE.SPR)
Contains: _QSF0KP0YP()                 (Params: none)
   Called by: ACTELSE.SPR
       Calls: ADDNEWELSE()             (function in ACTELSE.SPR)
       Calls: CLAUSE.SPR
       Calls: SETELSE                  (procedure in KBEDIT.SPR)
Contains: _QSF0KP13M()                 (Params: none)
   Called by: ACTELSE.SPR
       Calls: EDELSE                   (procedure in ACTELSE.SPR)
Contains: EDELSE                       (Params: none)
   Called by: _QSF0KP07D()             (function in ACTELSE.SPR)
   Called by: _QSF0KP0AR()             (function in ACTELSE.SPR)
   Called by: _QSF0KP0W3()             (function in ACTELSE.SPR)
   Called by: _QSF0KP13M()             (function in ACTELSE.SPR)
       Calls: CLAUSE.SPR
       Calls: SETELSE                  (procedure in KBEDIT.SPR)
Contains: ADDELSE                      (Params: none)
       Calls: CLAUSE.SPR
       Calls: SETELSE                  (procedure in KBEDIT.SPR)
Contains: ADDNEWELSE()                 (Params: none)
   Called by: ACTELSE.SPR
   Called by: _QSF0KP0YP()             (function in ACTELSE.SPR)
       Calls: CLAUSE.SPR
       Calls: SETELSE                  (procedure in KBEDIT.SPR)
```

11. CLAUSE.SPR

```
Contains: _QSF0KP3SZ()                 (Params: none)
   Called by: CLAUSE.SPR
Contains: _QSF0KP3X2()                 (Params: none)
   Called by: CLAUSE.SPR
Contains: _QSF0KP435()                 (Params: none)
   Called by: CLAUSE.SPR
Contains: _QSF0KP472()                 (Params: none)
   Called by: CLAUSE.SPR
Contains: _QSF0KP5HS()                 (Params: none)
   Called by: CLAUSE.SPR
Contains: _QSF0KP5LZ()                 (Params: none)
   Called by: CLAUSE.SPR
Contains: _QSF0KP5Q1()                 (Params: none)
   Called by: CLAUSE.SPR
Contains: _QSF0KP5UC()                 (Params: none)
   Called by: CLAUSE.SPR
Contains: INSNEWID()                   (Params: MDATA, MITEM)
   Called by: CLAUSE.SPR
       Calls: DP.PRG
Contains: SETOBJECT()                  (Params: M.OBJECT, MTYPE)
   Called by: CLAUSE.SPR
```

12. RULE.SPR

```
Contains: _QSF0KP97Y()                 (Params: none)
   Called by: RULE.SPR
```

```
Contains: _QSF0KPA3B()                 (Params: none)
   Called by: RULE.SPR
Contains: _QSF0KPA7J()                 (Params: none)
   Called by: RULE.SPR
       Calls: YESNO.PRG
```

13. TERM.SPR

```
Contains: _QSF0KPC3R()                 (Params: none)
   Called by: TERM.SPR
       Calls: EDPREM                   (procedure in TERM.SPR)
Contains: _QSF0KPC6D()                 (Params: none)
   Called by: TERM.SPR
       Calls: EDPREM                   (procedure in TERM.SPR)
Contains: _QSF0KPCPM()                 (Params: none)
   Called by: TERM.SPR
Contains: _QSF0KPCU9()                 (Params: none)
   Called by: TERM.SPR
       Calls: SETJOIN()                (function in TERM.SPR)
Contains: _QSF0KPCWU()                 (Params: none)
   Called by: TERM.SPR
       Calls: SETJOIN()                (function in TERM.SPR)
Contains: _QSF0KPCZG()                 (Params: none)
   Called by: TERM.SPR
       Calls: EDPREM                   (procedure in TERM.SPR)
Contains: _QSF0KPD1Z()                 (Params: none)
   Called by: TERM.SPR
       Calls: ADDFACT()                (function in TERM.SPR)
Contains: _QSF0KPD51()                 (Params: none)
   Called by: TERM.SPR
       Calls: EDPREM                   (procedure in TERM.SPR)
Contains: EDPREM                       (Params: none)
   Called by: _QSF0KPC3R()             (function in TERM.SPR)
   Called by: _QSF0KPC6D()             (function in TERM.SPR)
   Called by: _QSF0KPCZG()             (function in TERM.SPR)
   Called by: _QSF0KPD51()             (function in TERM.SPR)
       Calls: CLAUSE.SPR
       Calls: SETPREM                  (procedure in KBEDIT.SPR)
Contains: SETJOIN()                    (Params: JOINOP)
   Called by: _QSF0KPCU9()             (function in TERM.SPR)
   Called by: _QSF0KPCWU()             (function in TERM.SPR)
       Calls: SETPREM                  (procedure in KBEDIT.SPR)
Contains: LASTFACT()                   (Params: none)
Contains: ADDFACT()                    (Params: none)
   Called by: TERM.SPR
   Called by: _QSF0KPD1Z()             (function in TERM.SPR)
       Calls: CLAUSE.SPR
       Calls: SETPREM                  (procedure in KBEDIT.SPR)
```

14. OBJECT.SPR

```
Contains: _QSF0KPFR1()                 (Params: none)
   Called by: OBJECT.SPR
Contains: _QSF0KPFV8()                 (Params: none)
   Called by: OBJECT.SPR
       Calls: GETOBJ()                 (function in OBJECT.SPR)
Contains: _QSF0KPFYY()                 (Params: none)
```

```
      Called by: OBJECT.SPR
   Contains: _QSF0KPG47()                    (Params: none)
      Called by: OBJECT.SPR
          Calls: CREATARRAY()                (function in OBJECT.SPR)
          Calls: SELITEM()                   (function in OBJECT.SPR)
   Contains: _QSF0KPGOJ()                    (Params: none)
      Called by: OBJECT.SPR
   Contains: _QSF0KPGSR()                    (Params: none)
      Called by: OBJECT.SPR
          Calls: GETOBJ()                    (function in OBJECT.SPR)
   Contains: _QSF0KPGVZ()                    (Params: none)
      Called by: OBJECT.SPR
   Contains: _QSF0KPH17()                    (Params: none)
      Called by: OBJECT.SPR
          Calls: CREATARRAY()                (function in OBJECT.SPR)
          Calls: SELITEM()                   (function in OBJECT.SPR)
   Contains: CREATARRAY()                    (Params: none)
      Called by: _QSF0KPG47()                (function in OBJECT.SPR)
      Called by: _QSF0KPH17()                (function in OBJECT.SPR)
      Called by: GETOBJ()                    (function in OBJECT.SPR)
   Contains: DATAINPUT()                     (Params: none)
          Calls: DISEASE.SPR
          Calls: DICT.SPR
   Contains: GETOBJ()                        (Params: M.NAME)
      Called by: _QSF0KPFV8()                (function in OBJECT.SPR)
      Called by: _QSF0KPGSR()                (function in OBJECT.SPR)
          Calls: CREATARRAY()                (function in OBJECT.SPR)
          Calls: SELITEM()                   (function in OBJECT.SPR)
   Contains: SELITEM()                       (Params: none)
      Called by: _QSF0KPG47()                (function in OBJECT.SPR)
      Called by: _QSF0KPH17()                (function in OBJECT.SPR)
      Called by: GETOBJ()                    (function in OBJECT.SPR)
          Calls: OBLIST.SPR
          Calls: DISEASE.SPR
          Calls: DICT.SPR
```

15. ENUM.SPR

```
   Contains: _QSF0KPJAM()                    (Params: none)
      Called by: ENUM.SPR
   Contains: _QSF0KPJF7()                    (Params: none)
      Called by: ENUM.SPR
   Contains: _QSF0KPJJK()                    (Params: none)
      Called by: ENUM.SPR
   Contains: _QSF0KPJYY()                    (Params: none)
      Called by: ENUM.SPR
   Contains: _QSF0KPK3S()                    (Params: none)
      Called by: ENUM.SPR
   Contains: _QSF0KPK8E()                    (Params: none)
      Called by: ENUM.SPR
```

16. RESTORE.SPR

```
   Contains: _QSF0KPMPZ()                    (Params: none)
      Called by: RESTORE.SPR
   Contains: _QSF0KPMS0()                    (Params: none)
      Called by: RESTORE.SPR
```

```
            Calls: YESNO.PRG
            Calls: NEWPATHPOP              (procedure in BACKUP.SPR)
            Calls: NEWFILEPOP              (procedure in BACKUP.SPR)
    Contains: _QSF0KPMXI()                 (Params: none)
       Called by: RESTORE.SPR
            Calls: PATHSTRING              (procedure in BACKUP.SPR)
            Calls: NEWPATHPOP              (procedure in BACKUP.SPR)
            Calls: NEWFILEPOP              (procedure in BACKUP.SPR)
    Contains: _QSF0KPN0J()                 (Params: none)
       Called by: RESTORE.SPR
            Calls: PKUNZIP                 (procedure in RESTORE.SPR)
    Contains: _QSF0KPN3T()                 (Params: none)
       Called by: RESTORE.SPR
            Calls: NEWPATHPOP              (procedure in BACKUP.SPR)
            Calls: NEWFILEPOP              (procedure in BACKUP.SPR)
    Contains: _QSF0KPN7E()                 (Params: none)
       Called by: RESTORE.SPR
    Contains: _QSF0KPNY5()                 (Params: none)
       Called by: RESTORE.SPR
    Contains: _QSF0KPO0A()                 (Params: none)
       Called by: RESTORE.SPR
            Calls: YESNO.PRG
            Calls: NEWPATHPOP              (procedure in BACKUP.SPR)
            Calls: NEWFILEPOP              (procedure in BACKUP.SPR)
    Contains: _QSF0KPO5T()                 (Params: none)
       Called by: RESTORE.SPR
            Calls: PATHSTRING              (procedure in BACKUP.SPR)
            Calls: NEWPATHPOP              (procedure in BACKUP.SPR)
            Calls: NEWFILEPOP              (procedure in BACKUP.SPR)
    Contains: _QSF0KPO8T()                 (Params: none)
       Called by: RESTORE.SPR
            Calls: PKUNZIP                 (procedure in RESTORE.SPR)
    Contains: _QSF0KPOBY()                 (Params: none)
       Called by: RESTORE.SPR
            Calls: NEWPATHPOP              (procedure in BACKUP.SPR)
            Calls: NEWFILEPOP              (procedure in BACKUP.SPR)
    Contains: _QSF0KPOFG()                 (Params: none)
       Called by: RESTORE.SPR
    Contains: NEWDRIVEPOP                  (Params: none)
       Called by: BACKUP.SPR
       Called by: RESTORE.SPR
    Contains: NEWPATHPOP                   (Params: none)
       Called by: BACKUP.SPR
       Called by: RESTORE.SPR
       Called by: _QSF0KO98Q()             (function in BACKUP.SPR)
       Called by: _QSF0KO9EC()             (function in BACKUP.SPR)
       Called by: _QSF0KO9LV()             (function in BACKUP.SPR)
       Called by: _QSF0KOAID()             (function in BACKUP.SPR)
       Called by: _QSF0KOANU()             (function in BACKUP.SPR)
       Called by: _QSF0KOAUY()             (function in BACKUP.SPR)
       Called by: _QSF0KPMS0()             (function in RESTORE.SPR)
       Called by: _QSF0KPMXI()             (function in RESTORE.SPR)
       Called by: _QSF0KPN3T()             (function in RESTORE.SPR)
       Called by: _QSF0KPO0A()             (function in RESTORE.SPR)
       Called by: _QSF0KPO5T()             (function in RESTORE.SPR)
       Called by: _QSF0KPOBY()             (function in RESTORE.SPR)
    Contains: NEWFILEPOP                   (Params: none)
```

```
      Called by: BACKUP.SPR
      Called by: RESTORE.SPR
      Called by: _QSF0KO98Q()          (function in BACKUP.SPR)
      Called by: _QSF0KO9EC()          (function in BACKUP.SPR)
      Called by: _QSF0KO9LV()          (function in BACKUP.SPR)
      Called by: _QSF0KOAID()          (function in BACKUP.SPR)
      Called by: _QSF0KOANU()          (function in BACKUP.SPR)
      Called by: _QSF0KOAUY()          (function in BACKUP.SPR)
      Called by: _QSF0KPMS0()          (function in RESTORE.SPR)
      Called by: _QSF0KPMXI()          (function in RESTORE.SPR)
      Called by: _QSF0KPN3T()          (function in RESTORE.SPR)
      Called by: _QSF0KPO0A()          (function in RESTORE.SPR)
      Called by: _QSF0KPO5T()          (function in RESTORE.SPR)
      Called by: _QSF0KPOBY()          (function in RESTORE.SPR)
   Contains: PATHSTRING                (Params: none)
      Called by: _QSF0KO9EC()          (function in BACKUP.SPR)
      Called by: _QSF0KOANU()          (function in BACKUP.SPR)
      Called by: PKZIP                 (procedure in BACKUP.SPR)
      Called by: _QSF0KPMXI()          (function in RESTORE.SPR)
      Called by: _QSF0KPO5T()          (function in RESTORE.SPR)
      Called by: PKUNZIP               (procedure in RESTORE.SPR)
   Contains: PKUNZIP                   (Params: none)
      Called by: _QSF0KPN0J()          (function in RESTORE.SPR)
      Called by: _QSF0KPO8T()          (function in RESTORE.SPR)
          Calls: PATHSTRING            (procedure in BACKUP.SPR)
```

17. DD.SPR

```
   Contains: VALIDOBJ()                (Params: none)
      Called by: _QSF0KON7E()          (function in QUAL.SPR)
      Called by: _QSF0KPT2I()          (function in DD.SPR)
          Calls: ERRMSG.PRG
   Contains: QUALDEL()                 (Params: none)
      Called by: _QSF0KON7E()          (function in QUAL.SPR)
      Called by: _QSF0KPT2I()          (function in DD.SPR)
          Calls: POPUPSHOW()           (function in KBINIT.PRG)
          Calls: POPUPHIDE()           (function in KBINIT.PRG)
          Calls: SETQUALS              (procedure in KB.SPR)
   Contains: RESETDATA()               (Params: none)
      Called by: _QSF0KPT2I()          (function in DD.SPR)
   Contains: _QSF0KPT2I()              (Params: none)
      Called by: DD.SPR
          Calls: DDEDIT.SPR
          Calls: VALIDOBJ()            (function in QUAL.SPR)
          Calls: QUALDEL()             (function in QUAL.SPR)
          Calls: RESETDATA()           (function in DD.SPR)
   Contains: _QSF0KPT7D()              (Params: none)
      Called by: DD.SPR
          Calls: DDEDIT.SPR
```

18. PLAN.SPR

```
   Contains: _QSF0KPV8R()              (Params: none)
      Called by: PLAN.SPR
          Calls: OBJECT.SPR
          Calls: SETGOALS              (procedure in KB.SPR)
   Contains: _QSF0KPVCR()                  (Params: none)
      Called by: PLAN.SPR
          Calls: EDGOAL                    (procedure in KB.SPR)
   Contains: _QSF0KPVQB()                  (Params: none)
      Called by: PLAN.SPR
          Calls: OBJECT.SPR
          Calls: SETGOALS                  (procedure in KB.SPR)
   Contains: _QSF0KPVW8()                  (Params: none)
      Called by: PLAN.SPR
          Calls: EDGOAL                    (procedure in KB.SPR)
```

19. DICT.SPR

```
   Contains: EDENUM                        (Params: none)
      Called by: _QSF0KPY8W()              (function in DICT.SPR)
      Called by: _QSF0KQ6T0()              (function in DDEDIT.SPR)
          Calls: ENUM.SPR
          Calls: SETENUM                   (procedure in DICT.SPR)
          Calls: DDENUM.SPR
   Contains: SETENUM                       (Params: none)
      Called by: DICT.SPR
      Called by: DDEDIT.SPR
      Called by: EDENUM                    (procedure in DICT.SPR)
      Called by: _QSF0KQ6M1()              (function in DDEDIT.SPR)
   Contains: SETVAL                        (Params: none)
      Called by: _QSF0KQ6M1()              (function in DDEDIT.SPR)
   Contains: _QSF0KPY4X()                  (Params: none)
      Called by: DICT.SPR
   Contains: _QSF0KPY8W()                  (Params: none)
      Called by: DICT.SPR
          Calls: EDENUM                    (procedure in DICT.SPR)
   Contains: _QSF0KPYF6()                  (Params: none)
      Called by: DICT.SPR
   Contains: _QSF0KPYJ0()                  (Params: none)
      Called by: DICT.SPR
```

20. KBEDIT.SPR

```
   Contains: SETPREM                       (Params: none)
      Called by: KBEDIT.SPR
      Called by: _QSF0KOH4Z()              (function in KB.SPR)
      Called by: ADDFACT()                 (function in TERM.SPR)
      Called by: EDPREM                    (procedure in TERM.SPR)
      Called by: SETJOIN()                 (function in TERM.SPR)
      Called by: _QSF0KQ0SA()              (function in KBEDIT.SPR)
      Called by: _QSF0KQ14Z()              (function in KBEDIT.SPR)
          Calls: SEMPTY()                  (function in KBEDIT.SPR)
          Calls: HLINK()                   (function in KBEDIT.SPR)
          Calls: VLINK()                   (function in KBEDIT.SPR)
   Contains: VLINK()                       (Params: FROM, TO)
      Called by: SETPREM                   (procedure in KBEDIT.SPR)
   Contains: HLINK()                       (Params: S)
      Called by: SETPREM                   (procedure in KBEDIT.SPR)
   Contains: PUSH()                        (Params: N)
   Contains: POP()                         (Params: none)
   Contains: SEMPTY()                      (Params: none)
      Called by: SETPREM                   (procedure in KBEDIT.SPR)
```

```
Contains: SETACT                        (Params: none)
   Called by: KBEDIT.SPR
   Called by: _QSF0KOH4Z()              (function in KB.SPR)
   Called by: ADDNEWACT()               (function in ACTION.SPR)
   Called by: _QSF0KOY8A()              (function in ACTION.SPR)
   Called by: EDACT                     (procedure in ACTION.SPR)
   Called by: _QSF0KQ14Z()              (function in KBEDIT.SPR)
Contains: SETELSE                       (Params: none)
   Called by: KBEDIT.SPR
   Called by: ADDNEWELSE()              (function in ACTELSE.SPR)
   Called by: _QSF0KP0YP()              (function in ACTELSE.SPR)
   Called by: EDELSE                    (procedure in ACTELSE.SPR)
   Called by: ADDELSE                   (procedure in ACTELSE.SPR)
   Called by: _QSF0KQ10R()              (function in KBEDIT.SPR)
   Called by: _QSF0KQ14Z()              (function in KBEDIT.SPR)
Contains: SETACTION()                   (Params: none)
   Called by: KBEDIT.SPR
   Called by: _QSF0KQ14Z()              (function in KBEDIT.SPR)
Contains: _QSF0KQ0SA()                  (Params: none)
   Called by: KBEDIT.SPR
       Calls: TERM.SPR
       Calls: SETPREM                   (procedure in KBEDIT.SPR)
Contains: _QSF0KQ0WK()                  (Params: none)
   Called by: KBEDIT.SPR
       Calls: ACTION.SPR
Contains: _QSF0KQ10R()                  (Params: none)
   Called by: KBEDIT.SPR
       Calls: ACTELSE.SPR
       Calls: SETELSE                   (procedure in KBEDIT.SPR)
Contains: _QSF0KQ14Z()                  (Params: none)
   Called by: KBEDIT.SPR
       Calls: RULE.SPR
       Calls: SETPREM                   (procedure in KBEDIT.SPR)
       Calls: SETACT                    (procedure in KBEDIT.SPR)
       Calls: SETELSE                   (procedure in KBEDIT.SPR)
       Calls: SETACTION()               (function in KBEDIT.SPR)
```

21. OBLIST.SPR

```
Contains: _QSF0KQ4M3()                  (Params: none)
   Called by: OBLIST.SPR
```

22. DDEDIT.SPR

```
Contains: SETRULES()                    (Params: none)
   Called by: DDEDIT.SPR
       Calls: AREANAME()                (function in DDEDIT.SPR)
Contains: EDENUM                        (Params: none)
   Called by: _QSF0KPY8W()              (function in DICT.SPR)
   Called by: _QSF0KQ6T0()              (function in DDEDIT.SPR)
       Calls: ENUM.SPR
       Calls: SETENUM                   (procedure in DICT.SPR)
       Calls: DDENUM.SPR
Contains: SETENUM                       (Params: none)
   Called by: DICT.SPR
   Called by: DDEDIT.SPR
   Called by: EDENUM                    (procedure in DICT.SPR)
   Called by: _QSF0KQ6M1()              (function in DDEDIT.SPR)
Contains: SETVAL                        (Params: none)
   Called by: _QSF0KQ6M1()              (function in DDEDIT.SPR)
Contains: CLEANENUM()                   (Params: none)
   Called by: _QSF0KQ6YY()              (function in DDEDIT.SPR)
Contains: AREANAME()                    (Params: MID)
   Called by: SETRULES()                (function in DDEDIT.SPR)
Contains: _QSF0KQ6M1()                  (Params: none)
   Called by: DDEDIT.SPR
       Calls: SETVAL                    (procedure in DICT.SPR)
       Calls: SETENUM                   (procedure in DICT.SPR)
       Calls: DDENUM.SPR
Contains: _QSF0KQ6T0()                  (Params: none)
   Called by: DDEDIT.SPR
       Calls: EDENUM                    (procedure in DICT.SPR)
Contains: _QSF0KQ6YY()                  (Params: none)
   Called by: DDEDIT.SPR
       Calls: CLEANENUM()               (function in DDEDIT.SPR)
Contains: _QSF0KQ74F()                  (Params: none)
   Called by: DDEDIT.SPR
```

23. DDENUM.SPR

```
Contains: _QSF0KQ9MB()                  (Params: none)
   Called by: DDENUM.SPR
Contains: _QSF0KQ9QC()                  (Params: none)
   Called by: DDENUM.SPR
```

24. BACKUP.SPR

```
Contains: _QSF0KO96C()                  (Params: none)
   Called by: BACKUP.SPR
Contains: _QSF0KO98Q()                  (Params: none)
   Called by: BACKUP.SPR
       Calls: YESNO.PRG
       Calls: NEWPATHPOP                (procedure in BACKUP.SPR)
       Calls: NEWFILEPOP                (procedure in BACKUP.SPR)
Contains: _QSF0KO9EC()                  (Params: none)
   Called by: BACKUP.SPR
       Calls: PATHSTRING                (procedure in BACKUP.SPR)
       Calls: NEWPATHPOP                (procedure in BACKUP.SPR)
       Calls: NEWFILEPOP                (procedure in BACKUP.SPR)
Contains: _QSF0KO9HK()                  (Params: none)
   Called by: BACKUP.SPR
       Calls: YESNO.PRG
       Calls: PKZIP                     (procedure in BACKUP.SPR)
Contains: _QSF0KO9LV()                  (Params: none)
   Called by: BACKUP.SPR
       Calls: NEWPATHPOP                (procedure in BACKUP.SPR)
       Calls: NEWFILEPOP                (procedure in BACKUP.SPR)
Contains: _QSF0KOAG9()                  (Params: none)
   Called by: BACKUP.SPR
Contains: _QSF0KOAID()                  (Params: none)
   Called by: BACKUP.SPR
       Calls: YESNO.PRG
       Calls: NEWPATHPOP                (procedure in BACKUP.SPR)
       Calls: NEWFILEPOP                (procedure in BACKUP.SPR)
```

```
Contains: _QSF0KOANU()                  (Params: none)
   Called by: BACKUP.SPR
       Calls: PATHSTRING                (procedure in BACKUP.SPR)
       Calls: NEWPATHPOP                (procedure in BACKUP.SPR)
       Calls: NEWFILEPOP                (procedure in BACKUP.SPR)
Contains: _QSF0KOAQV()                  (Params: none)
   Called by: BACKUP.SPR
       Calls: YESNO.PRG
       Calls: PKZIP                     (procedure in BACKUP.SPR)
Contains: _QSF0KOAUY()                  (Params: none)
   Called by: BACKUP.SPR
       Calls: NEWPATHPOP                (procedure in BACKUP.SPR)
       Calls: NEWFILEPOP                (procedure in BACKUP.SPR)
Contains: NEWDRIVEPOP                   (Params: none)
   Called by: BACKUP.SPR
   Called by: RESTORE.SPR
Contains: NEWPATHPOP                    (Params: none)
   Called by: BACKUP.SPR
   Called by: RESTORE.SPR
   Called by: _QS*                      (function in BACKUP.SPR)
Contains: NEWFILEPOP                    (Params: none)
   Called by: BACKUP.SPR
   Called by: RESTORE.SPR
   Called by: _QSF0KO98Q()              (function in BACKUP.SPR)
   Called by: _QSF0KOAUY()              (function in BACKUP.SPR)
   Called by: _QSF0KPOBY()              (function in RESTORE.SPR)
Contains: PATHSTRING                    (Params: none)
   Called by: _QSF0KO9EC()              (function in BACKUP.SPR)
   Called by: _QSF0KOANU()              (function in BACKUP.SPR)
   Called by: PKZIP                     (procedure in BACKUP.SPR)
   Called by: _QSF0KPMXI()              (function in RESTORE.SPR)
   Called by: _QSF0KPO5T()              (function in RESTORE.SPR)
   Called by: PKUNZIP                   (procedure RESTORE.SPR)
Contains: PKZIP                         (Params: none)
   Called by: _QSF0KO9HK()              (function in BACKUP.SPR)
   Called by: _QSF0KOAQV()              (function in BACKUP.SPR)
       Calls: PATHSTRING                (procedure in BACKUP.SPR)
```

Section VII. Program Source Code

```
*:***********************************************************************
=> *******
*:
*: Procedure file: C:\CAMD2\KBEDIT\WORKW\KBINIT.PRG
*:         System: Knowledge Base Editor
*:         Author: Hoa L. Ly
*:      Copyright (c) June 1, 1994, Naval Health Research Center, Code 2
=> 2
*:  Last modified: 08/05/94 at 14:05:34
*:
*:  Procs & Fncts: SETPATH()
*:               : MYHANDLER()
*:               : _QUIT()
*:               : POPUPSHOW()
*:               : POPUPHIDE()
*:
*:          Calls: SETPATH()           (function in KBINIT.PRG)
*:               : KBMENU.MPR
*:               : MYHANDLER()         (function in KBINIT.PRG)
*:
*:      Documented 15:00:46                                FoxDoc versio
=> n 3.00a
*:***********************************************************************
=> *******
    *********************************************************************
=> *
    *
    *    Initialize system init_app()
    *
    *********************************************************************
=> *
    CLOSE ALL
    CLEAR WINDOWS ALL
    CLEAR ALL
    PUBLIC m.pass,savetalk,savesc        && Initialize public variables
    PUBLIC dropdead, mproc
    PUBLIC m.new, m.home, m.data

  ┌─IF SET('TALK') = 'ON'     && TALK handled as a special case.
  │    SET TALK OFF          && Turn TALK OFF
  │    savetalk = 'ON'       && TALK was ON, save the setting
  ├─ELSE                      && TALK is OFF
  │    savetalk = 'OFF'      && TALK was OFF, save the setting
  └─ENDIF

  ┌─IF SET('ESCAPE') = 'ON'  && ESCAPE handled as a special case.
  │    SET ESCAPE OFF       && Turn ESCAPE OFF
  │    savesc = 'ON'        && ESCAPE was ON, save the setting
  ├─ELSE                     && ESCAPE is OFF
  │    savesc = 'OFF'       && ESCAPE was OFF, save the setting
  └─ENDIF

    && End of init_app()
    m.home = SYS(2003)
    m.data = GETENV("KBDATA")
  ┌─IF !EMPTY(m.data)
  │    DO setpath WITH m.data
  └─ENDIF
    m.bak = m.data + "\bak\"
    m.new = m.data + "\new\"

    *-------------------------------------------
    * Backup request.
    *-------------------------------------------
    *m.do = yesno("Do you want to Backup database?","YES","NO")
    *IF m.do
    *   do backup
    *ENDIF

    *setup help file
    SET HELP TO kbhelp.dbf
    ON KEY LABEL f1 HELP

    *-------------------------------------------
    * Install menu
    *-------------------------------------------
    dropdead = .F.
    PUSH MENU _msysmenu                  && HLL
    DO kbmenu.mpr         && Launch application menu
    READ VALID myhandler()


    POP MENU _msysmenu          && HLL

    *-------------------------------------------
    * Restore request.
    *-------------------------------------------
    CLEAR WINDOWS ALL
    CLOSE DATABASES
    SET TOPIC TO "restore"
    *m.do = yesno("Do you want to Restore?","YES","NO")
    *IF m.do
    *   do restore
    *ENDIF
    SET HELP TO

    *********************************************************************
=> *
    *     DONE
    *********************************************************************
=> *
    CLEAR WINDOWS ALL
    CLOSE ALL
<───RETURN

*!**********************************************************************
=> *******
*!
*!       Function: MYHANDLER
*!
*!      Called by: KBINIT.PRG
*!
*!**********************************************************************
=> *******
    FUNCTION myhandler
  ┌─IF TYPE("dropdead") = "U"
  │    dropdead = .F.
  └─ENDIF
<───RETURN dropdead

    *-----------------------------------------------------
    *
    * Quit the system
    *
    *-----------------------------------------------------

*!**********************************************************************
=> *******
*!
*!       Function: _QUIT
*!
*!      Called by: KBMENU.MPR
```

```
*!
*!          Calls: ERRMSG.PRG
*!
*!*****************************************************************************
=> *******
    FUNCTION _quit
   ┌IF EMPTY( WOUTPUT())
   │   dropdead = .T.
   │   CLEAR READ ALL
   ├ELSE
   │   =errmsg("Close windows before quitting",1)
   └ENDIF
<──RETURN

    *-------------------------------------------------------
    *
    * display popup notice
    *
    *-------------------------------------------------------

*!*****************************************************************************
=> *******
*!
*!       Function: POPUPSHOW
*!
*!      Called by: QUALDEL()          (function in QUAL.SPR)
*!               : _QSFOKQC6T()       (function in KBLOAD.SPR)
*!
*!*****************************************************************************
=> *******
    FUNCTION popupshow
    PARAMETERS errstr
   ┌IF NOT WEXIST("w_popnote")
   │   DEFINE WINDOW w_popnote ;
   │      FROM INT((SROW()-8)/2),INT((SCOL()-36)/2) ;
   │      TO INT((SROW()-8)/2)+7,INT((SCOL()-36)/2)+35 ;
   │      TITLE "One moment" ;
   │      FLOAT ;
   │      CLOSE ;
   │      SHADOW ;
   │      DOUBLE ;
   │      COLOR SCHEME 1
   └ENDIF
   ┌IF WVISIBLE("w_popnote")
   │   ACTIVATE WINDOW w_popnote SAME
   ├ELSE
   │   ACTIVATE WINDOW w_popnote NOSHOW
   └ENDIF
    @ 1,1 SAY errstr SIZE 3,31

   ┌IF NOT WVISIBLE("w_popnote")
   │   ACTIVATE WINDOW w_popnote
   └ENDIF
<──RETURN ""

*!*****************************************************************************
=> *******
*!
*!       Function: POPUPHIDE
*!
*!      Called by: QUALDEL()          (function in QUAL.SPR)
*!               : _QSFOKQC6T()       (function in KBLOAD.SPR)
*!
*!*****************************************************************************
=> *******
    FUNCTION popuphide
    RELEASE WINDOW w_popnote
<──RETURN

    *-------------------------------------------
    * Quit out the system
    *-------------------------------------------
    CLOSE DATABASES
    *set default to (m.home)

    *-------------------------------------------------------
    * Creat foxpro path
    *-------------------------------------------------------
*!*****************************************************************************
=> *******
*!
*!       Function: SETPATH
*!
*!      Called by: KBINIT.PRG
*!
*!*****************************************************************************
=> *******
    FUNCTION setpath
    PARAMETER newpath
    PRIVATE mnewpath
   ┌IF !EMPTY(newpath)
   │   mnewpath = FULLPATH(newpath) + ";"
   │   mcurpath = IIF(EMPTY(SET("PATH")),FULLPATH(CURDIR())+";",SET("PATH
=> "))
   │  ┌IF NOT mcurpath $ mnewpath
   │  │   mcurpath = mcurpath + mnewpath      && add sams subdirectory to
=>  path
   │  └ENDIF
   │   SET PATH TO (mcurpath)
   └ENDIF
<──RETURN .T.




*:  EOF: KBINIT.act
```

1C/14PRINT

```
*:************************************************************************
=> *******
*:
*: Procedure file: C:\CAMD2\KBEDIT\WORKW\YESNO.PRG
*:         System: Knowledge Base Editor
*:         Author: Hoa L. Ly
*:      Copyright (c) June 1, 1994, Naval Health Research Center, Code 2
=> 2
*:  Last modified: 08/05/94 at 11:38:50
*:
*:         Set by: _QSFOKO98Q()      (function in BACKUP.SPR)
*:               : _QSFOKO9HK()      (function in BACKUP.SPR)
*:               : _QSFOKOAID()      (function in BACKUP.SPR)
*:               : _QSFOKOAQV()      (function in BACKUP.SPR)
*:               : _QSFOKPA7J()      (function in RULE.SPR)
*:               : _QSFOKPMSO()      (function in RESTORE.SPR)
*:               : _QSFOKPOOA()      (function in RESTORE.SPR)
*:               : CHECKFILE()       (function in KBLOAD.SPR)
*:
*:      Documented 15:00:47                              FoxDoc versio
=> n 3.00a
*:************************************************************************
=> *******
   *       **********************************************************
   *       *                                                        *
   *       * 11/26/91             YESNO.PRG               02:50:49 *
   *       *                                                        *
   *       **********************************************************
   *       *                                                        *
   *       * Adam Green                                             *
   *       *                                                        *
   *       * Copyright (c) 1991 Adam Green Seminars                 *
   *       * One Faneuil Hall                                       *
   *       * Boston, MA  02174                                      *
   *       *                                                        *
   *       * Description:                                           *
   *       * This program was automatically generated by GENSCRN.   *
   *       *                                                        *
   *       **********************************************************


   *       **********************************************************
   *       *                                                        *
   *       *              YESNO Setup Code - SECTION 1              *
   *       *                                                        *
   *       **********************************************************
   *

   #REGION 1
   PARAMETERS MESSAGE, ok, CANCEL
   PRIVATE ALL
   *HLL, If message was empty print a plain window just only need respon
=> se
 ┌IF PARAMETERS() = 0
 │   * Default to plain prompts
 │   m.message = ""
 └ENDIF

   * Ok and Cancel are used for prompts in the push buttons
 ┌IF PARAMETERS() = 1
 │   * Default to normal prompts
 │   m.ok = "OK"
 │   m.cancel = "Cancel"
 └ENDIF

   * Truncate any message longer than 50 characters
```

```
┌IF LEN( m.message ) > 129
│   * The message is centered in an @SAY
│   m.message = SUBSTR( m.message, 1, 129 )
└ENDIF

 PUSH KEY CLEAR

 #REGION 0
 REGIONAL m.currarea, m.talkstat, m.compstat

┌IF SET("TALK") = "ON"
│   SET TALK OFF
│   m.talkstat = "ON"
├ELSE
│   m.talkstat = "OFF"
└ENDIF
 m.compstat = SET("COMPATIBLE")
 SET COMPATIBLE FOXPLUS

 *       **********************************************************
 *       *                                                        *
 *       *                 Window definitions                     *
 *       *                                                        *
 *       **********************************************************
 *

┌IF NOT WEXIST("yesno")
│   DEFINE WINDOW yesno ;
│      AT 0,0 ;
│      SIZE 7,72 ;
│      FONT "MS SANS SERIF", 8 ;
│      FLOAT ;
│      NOCLOSE ;
│      DOUBLE
│   MOVE WINDOW yesno CENTER
│
└ENDIF


 *       **********************************************************
 *       *                                                        *
 *       *                 YESNO Screen Layout                    *
 *       *                                                        *
 *       **********************************************************
 *

 #REGION 1
┌IF WVISIBLE("yesno")
│   ACTIVATE WINDOW yesno SAME
├ELSE
│   ACTIVATE WINDOW yesno NOSHOW
└ENDIF
 @ 1,2 SAY m.message ;
    FONT "MS Sans Serif", 8 ;
    SIZE 2,65
 @ 4,18 GET m.answer ;
    PICTURE "@*HT \!\<&OK;\<&Cancel" ;
    SIZE 1.5,11,4 ;
    FONT "MS Sans Serif", 8 ;
    DEFAULT 1;
    STYLE "B"

┌IF NOT WVISIBLE("yesno")
│   ACTIVATE WINDOW yesno
└ENDIF

```

```
    READ CYCLE MODAL

    RELEASE WINDOW yesno

    #REGION 0
  ┌─IF m.talkstat = "ON"
  │    SET TALK ON
  └─ENDIF
  ┌─IF m.compstat = "ON"
  │    SET COMPATIBLE ON
  └─ENDIF


    *       *********************************************************
    *       *                                                       *
    *       *                  YESNO Cleanup Code                   *
    *       *                                                       *
    *       *********************************************************
    *

    #REGION 1
    POP KEY

    * Convert the numeric value of m.Answer from 1|2 to .T.|.F.
    * If the user selected OK and didn't exit with Escape
   ┌─IF m.answer = 1 AND LASTKEY() <> 27
<══════RETURN .T.
   ├─ELSE
   │    * Cancel or Escape returns false
<══════RETURN .F.
   └─ENDIF
 *:  EOF: YESNO.act
```

```
*:**********************************************************************
=> *******
*:
*: Procedure file: C:\CAMD2\KBEDIT\WORKW\BACKUP.PRG
*:         System: Knowledge Base Editor
*:         Author: Hoa L. Ly
*:      Copyright (c) June 1, 1994, Naval Health Research Center, Code 2
=> 2
*:  Last modified: 02/15/94 at 13:31:20
*:
*:         Set by: KBMENU.MPR
*:
*:          Calls: BACKUP.SPR
*:
*:      Documented 15:00:47                                FoxDoc versio
=> n 3.00a
*:**********************************************************************
=> *******
    *-----------------------------------------------------------------
=> ----
    * Backup data files
    * Programmer: HLL
    *-----------------------------------------------------------------
=> ---
    * PROCEDURE backup
    PUBLIC mscreen
    PRIVATE MESSAGE
    SAVE SCREEN TO mscreen
    MESSAGE = "Backup Knowledge Base database"
    SET TOPIC TO "BACKUP"
    DO backup.spr WITH MESSAGE
    RESTORE SCREEN FROM mscreen
<---RETURN

*:  EOF: BACKUP.act
```

```
*
*
*     08/09/94              KBMENU.MPR              09:38:40
*
*
*
*     Author's Name
*
*     Copyright (c) 1994 Company Name
*     Address
*     City,     Zip
*
*     Description:
*     This program was automatically generated by GENMENU.
*
*


*
*
*                         Menu Definition
*
*
*

SET SYSMENU TO

SET SYSMENU AUTOMATIC

DEFINE PAD _qsf0ko6j4 OF _msysmenu PROMPT "System" COLOR SCHEME 3
DEFINE PAD _qsf0ko6je OF _msysmenu PROMPT "Knowledge Base" COLOR SCHE
=> ME 3
DEFINE PAD _qsf0ko6jm OF _msysmenu PROMPT "Exit system" COLOR SCHEME
=> 3
ON PAD _qsf0ko6j4 OF _msysmenu ACTIVATE POPUP SYSTEM
ON PAD _qsf0ko6je OF _msysmenu ACTIVATE POPUP knowledgeb
ON SELECTION PAD _qsf0ko6jm OF _msysmenu DO _quit

DEFINE POPUP SYSTEM MARGIN RELATIVE SHADOW COLOR SCHEME 4
DEFINE BAR _mst_help OF SYSTEM PROMPT "\<Help                 F1"
DEFINE BAR 2 OF SYSTEM PROMPT "\-"
DEFINE BAR 3 OF SYSTEM PROMPT "\<Backup"
DEFINE BAR 4 OF SYSTEM PROMPT "\<Restore"
ON SELECTION BAR 3 OF SYSTEM DO backup
ON SELECTION BAR 4 OF SYSTEM DO RESTORE

DEFINE POPUP knowledgeb MARGIN RELATIVE SHADOW COLOR SCHEME 4
DEFINE BAR 1 OF knowledgeb PROMPT "Knowledge Base Area"
DEFINE BAR 2 OF knowledgeb PROMPT "Question - Data Dictionary"
ON SELECTION BAR 1 OF knowledgeb DO kb.spr
ON SELECTION BAR 2 OF knowledgeb DO dd.spr
*:  EOF: KBMENU.ac2
```

```
*
*       08/09/94            BACKUP.SPR             09:38:42
*
*
*       Author's Name
*
*       Copyright (c) 1994 Company Name
*       Address
*       City,     Zip
*
*       Description:
*       This program was automatically generated by GENSCRN.
*
*
*

PARAMETERS MESSAGE, wildcard, filename
DO CASE
CASE _WINDOWS


     *
     *
     *            BACKUP/Windows Setup Code - SECTION 1
     *
     *
     *

     #REGION 1
     PRIVATE mfanme, mpath, olddrive, predrive, oldpath, maction, mdrive, mfiles
     DO CASE
     CASE PARAMETERS() = 0
        m.message = ""
        m.wildcard = "*.ZIP"
        mfname = "BU" + STRTRAN(DTOC(DATE()),"/","")
     CASE PARAMETERS() = 1 OR EMPTY(m.wildcard)
        m.wildcard = "*.ZIP"
        mfname = "BU" + STRTRAN(DTOC(DATE()),"/","")
     CASE PARAMETERS() = 2 OR EMPTY(m.filename)
        mfname = "BU" + STRTRAN(DTOC(DATE()),"/","")
     OTHERWISE
        mfname = filename
     ENDCASE


     #REGION 0
     REGIONAL m.currarea, m.talkstat, m.compstat

     IF SET("TALK") = "ON"
        SET TALK OFF
        m.talkstat = "ON"
     ELSE
        m.talkstat = "OFF"
     ENDIF
     m.compstat = SET("COMPATIBLE")
     SET COMPATIBLE FOXPLUS

     m.rborder = SET("READBORDER")
     SET readborder ON

     *
     *
     *            Windows Window definitions
     *
     *
     *

     IF NOT WEXIST("w_backup") ;
           OR UPPER(WTITLE("W_BACKUP")) == "W_BACKUP.PJX" ;
           OR UPPER(WTITLE("W_BACKUP")) == "W_BACKUP.SCX" ;
           OR UPPER(WTITLE("W_BACKUP")) == "W_BACKUP.MNX" ;
           OR UPPER(WTITLE("W_BACKUP")) == "W_BACKUP.PRG" ;
           OR UPPER(WTITLE("W_BACKUP")) == "W_BACKUP.FRX" ;
           OR UPPER(WTITLE("W_BACKUP")) == "W_BACKUP.QPR"
        DEFINE WINDOW w_backup ;
           AT  0.000, 0.000  ;
           SIZE 22.167,52.500 ;
           FONT "Terminal", 8 ;
           NOFLOAT ;
           NOCLOSE ;
           SHADOW ;
           NOMINIMIZE ;
           DOUBLE
        MOVE WINDOW w_backup CENTER
     ENDIF


     *
     *
     *            BACKUP/Windows Setup Code - SECTION 2
     *
     *
     *

     #REGION 1
     m.olddrive = SET("DEFAULT")
     m.prevdrive = 1
     m.oldpath = CURDIR()
     maction = 1

     DECLARE drivearray[1,1]
     mdrive = 1
     DO newdrivepop

     DECLARE patharray[1,1]
     mpath = 1
     DO newpathpop

     DECLARE filearray[1,1]
     mfiles = 1
     DO newfilepop


     *=closefiles()

```

```




    *
    *
    *        BACKUP/Windows Screen Layout
    *
    *
    *

    #REGION 1
    IF WVISIBLE("w_backup")
       ACTIVATE WINDOW w_backup SAME
    ELSE
       ACTIVATE WINDOW w_backup NOSHOW
    ENDIF
    @ 5.083,26.375 SAY "To drive: "  ;
       FONT "Terminal", 8
    @ 4.917,37.375 GET mdrive ;
       PICTURE "@^" ;
       FROM drivearray ;
       SIZE 1.500,12.500 ;
       DEFAULT 1 ;
       FONT "Terminal", 8 ;
       WHEN _qsf0ko96c() ;
       VALID _qsf0ko98q()
    @ 8.667,26.125 SAY "Directory:"  ;
       FONT "Terminal", 8
    @ 0.917,3.125 SAY MESSAGE ;
       SIZE 1.333,42.500 ;
       FONT "Terminal", 8
    @ 8.333,37.500 GET mpath ;
       PICTURE "@^" ;
       FROM patharray ;
       SIZE 1.500,12.500 ;
       DEFAULT 1 ;
       FONT "Terminal", 8 ;
       VALID _qsf0ko9ec()
    @ 12.917,26.500 GET maction ;
       PICTURE "@*HT \!\<Backup;\<Cancel" ;
       SIZE 2.083,9.625,2.500 ;
       DEFAULT 1 ;
       FONT "Terminal", 8 ;
       VALID _qsf0ko9hk()
    @ 3.750,3.000 GET mfile ;
       PICTURE "@&N" ;
       FROM filearray ;
       SIZE 14.000,21.375 ;
       DEFAULT 1 ;
       FONT "Terminal", 8 ;
       VALID _qsf0ko9lv()
    @ 18.667,3.125 SAY "Backup file name: "  ;
       FONT "Terminal", 8
    @ 18.583,22.375 GET mfname ;
       SIZE 1.083,27.000 ;
       DEFAULT " " ;
       FONT "Terminal", 8

    IF NOT WVISIBLE("w_backup")
       ACTIVATE WINDOW w_backup
    ENDIF

    READ CYCLE MODAL

    RELEASE WINDOW w_backup

    #REGION 0

    SET readborder &rborder

    IF m.talkstat = "ON"
       SET TALK ON
    ENDIF
    IF m.compstat = "ON"
       SET COMPATIBLE ON
    ENDIF


     *
     *
     *        BACKUP/Windows Cleanup Code
     *
     *
     *

     #REGION 1
     SET DEFAULT TO (m.olddrive + m.oldpath)
     RETURN

     *****************************************************

     ********************

CASE _DOS


     *
     *
     *        BACKUP/MS-DOS Setup Code - SECTION 1
     *
     *
     *

     #REGION 1
     PRIVATE mfanme, mpath, olddrive, predrive, oldpath, maction, mdrive, mfiles
     DO CASE
     CASE PARAMETERS() = 0
        m.message = ""
        m.wildcard = "*.ZIP"
        mfname = "BU" + STRTRAN(DTOC(DATE()),"/","")
     CASE PARAMETERS() = 1 OR EMPTY(m.wildcard)
        m.wildcard = "*.ZIP"
        mfname = "BU" + STRTRAN(DTOC(DATE()),"/","")
     CASE PARAMETERS() = 2 OR EMPTY(m.filename)
        mfname = "BU" + STRTRAN(DTOC(DATE()),"/","")
```

```
 OTHERWISE
    mfname = filename
 ENDCASE


 #REGION 0
 REGIONAL m.currarea, m.talkstat, m.compstat

 IF SET("TALK") = "ON"
    SET TALK OFF
    m.talkstat = "ON"
 ELSE
    m.talkstat = "OFF"
 ENDIF
 m.compstat = SET("COMPATIBLE")
 SET COMPATIBLE FOXPLUS

 *
 *
 *             MS-DOS Window definitions
 *
 *
 *

 IF NOT WEXIST("w_backup") ;
       OR UPPER(WTITLE("W_BACKUP")) == "W_BACKUP.PJX" ;
       OR UPPER(WTITLE("W_BACKUP")) == "W_BACKUP.SCX" ;
       OR UPPER(WTITLE("W_BACKUP")) == "W_BACKUP.MNX" ;
       OR UPPER(WTITLE("W_BACKUP")) == "W_BACKUP.PRG" ;
       OR UPPER(WTITLE("W_BACKUP")) == "W_BACKUP.FRX" ;
       OR UPPER(WTITLE("W_BACKUP")) == "W_BACKUP.QPR"
    DEFINE WINDOW w_backup ;
       FROM INT((SROW()-18)/2),INT((SCOL()-61)/2) ;
       TO INT((SROW()-18)/2)+17,INT((SCOL()-61)/2)+60 ;
       NOFLOAT ;
       NOCLOSE ;
       SHADOW ;
       NOMINIMIZE ;
       DOUBLE ;
       COLOR SCHEME 5
 ENDIF


 *
 *
 *             BACKUP/MS-DOS Setup Code - SECTION 2
 *
 *
 *

 #REGION 1
 m.olddrive = SET("DEFAULT")
 m.prevdrive = 1
 m.oldpath = CURDIR()
 maction = 1

 DECLARE drivearray[1,1]
 mdrive = 1
 DO newdrivepop

 DECLARE patharray[1,1]
 mpath = 1
 DO newpathpop

 DECLARE filearray[1,1]
 mfiles = 1
 DO newfilepop


 *=closefiles()






 *
 *
 *             BACKUP/MS-DOS Screen Layout
 *
 *
 *

 #REGION 1
 IF WVISIBLE("w_backup")
    ACTIVATE WINDOW w_backup SAME
 ELSE
    ACTIVATE WINDOW w_backup NOSHOW
 ENDIF
 @ 3,29 SAY "To drive: " ;
    SIZE 1,10, 0
 @ 2,40 GET mdrive ;
    PICTURE "@^" ;
    FROM drivearray ;
    SIZE 3,18 ;
    DEFAULT 1 ;
    WHEN _qsf0koag9() ;
    VALID _qsf0koaid() ;
    COLOR SCHEME 5, 6
 @ 6,29 SAY "Directory:" ;
    SIZE 1,10, 0
 @ 0,1 SAY MESSAGE ;
    SIZE 1,40
 @ 5,40 GET mpath ;
    PICTURE "@^" ;
    FROM patharray ;
    SIZE 3,18 ;
    DEFAULT 1 ;
    VALID _qsf0koanu() ;
    COLOR SCHEME 5, 6
 @ 9,45 GET maction ;
    PICTURE "@*VT \!\<Backup;\<Cancel" ;
    SIZE 1,8,1 ;
    DEFAULT 1 ;
    VALID _qsf0koaqv()
 @ 2,1 GET mfile ;
    PICTURE "@&N" ;
    FROM filearray ;
```

```
           SIZE 11,26 ;
           DEFAULT 1 ;
           VALID _qsf0koauy() ;
           COLOR SCHEME 6
        @ 14,1 SAY "Backup file name: " ;
           SIZE 1,18, 0
        @ 14,19 GET mfname ;
           SIZE 1,39 ;
           DEFAULT " "

        IF NOT WVISIBLE("w_backup")
           ACTIVATE WINDOW w_backup
        ENDIF

        READ CYCLE MODAL

        RELEASE WINDOW w_backup

        #REGION 0
        IF m.talkstat = "ON"
           SET TALK ON
        ENDIF
        IF m.compstat = "ON"
           SET COMPATIBLE ON
        ENDIF


         *
=>
         *
=>
         *          BACKUP/MS-DOS Cleanup Code
=>
         *
=>
         *
=>
         *

         #REGION 1
         SET DEFAULT TO (m.olddrive + m.oldpath)
         RETURN

         ******************************************************

         *********************

      ENDCASE


      *
      *
      *      _QSF0K096C          mdrive WHEN
      *
      *      Function Origin:
      *
      *      From Platform:      Windows
      *      From Screen:        BACKUP,     Record Number:    3
      *      Variable:           mdrive
      *      Called By:          WHEN Clause
      *      Object Type:        Popup
      *      Snippet Number:     1
      *
      *
      *
      FUNCTION _qsf0ko96c     &&  mdrive WHEN
      #REGION 1
      m.prevdrive = mdrive

      *
      *
      *      _QSF0K098Q          mdrive VALID
      *
      *      Function Origin:
      *
      *      From Platform:      Windows
      *      From Screen:        BACKUP,     Record Number:    3
      *      Variable:           mdrive
      *      Called By:          VALID Clause
      *      Object Type:        Popup
      *      Snippet Number:     2
      *
      *
      *
      *Switch to the selected drive
      FUNCTION _qsf0ko98q     &&  mdrive VALID
      #REGION 1
      PRIVATE newdrive,mready

      *Convert the popup bar number into the matching drive name
      m.newdrive = drivearray[mdrive]

      IF UPPER(m.newdrive) $ "A:B:"
         mready = yesno("Please insert disk into drive " + m.newdrive, "rea
=> dy","cancel")
      ELSE
         mready = .T.
      ENDIF
      IF mready
         *Go there and reset all the other popups to match
         SET DEFAULT TO (m.newdrive)
         DO newpathpop
         DO newfilepop
      ELSE
         mdrive =  m.prevdrive
         m.newdrive = drivearray[mdrive]
      ENDIF
       SHOW GETS

       *
       *
       *      _QSF0K09EC          mpath VALID
       *
       *      Function Origin:
       *
       *      From Platform:      Windows
       *      From Screen:        BACKUP,     Record Number:    6
       *      Variable:           mpath
       *      Called By:          VALID Clause
       *      Object Type:        Popup
       *      Snippet Number:     3
       *
       *
       *
       FUNCTION _qsf0ko9ec     &&  mpath VALID
       #REGION 1
       m.newdefault = pathstring()
       SET DEFAULT TO (m.newdefault)
       DO newpathpop
       DO newfilepop
       SHOW GETS
```

D-4

```
*
*
*       _QSFOKO9HK          maction VALID
*
*       Function Origin:
*
*       From Platform:      Windows
*       From Screen:        BACKUP,     Record Number:    7
*       Variable:           maction
*       Called By:          VALID Clause
*       Object Type:        Push Button
*       Snippet Number:     4
*
*
*
FUNCTION _qsf0ko9hk     &&  maction VALID
#REGION 1
DO CASE
CASE maction = 1

   pkzip = .T.
   IF FILE(mfname)
      mdel = yesno( mfname + " already exists, overwrite it? ", "Yes"
=> , "Cancel")
      IF mdel
         DELETE FILE &mfname
      ELSE
         pkzip = .F.
      ENDIF
   ENDIF
   IF pkzip
      DO pkzip
   ENDIF
OTHERWISE
ENDCASE

*
*
*       _QSFOKO9LV          mfile VALID
*
*       Function Origin:
*
*       From Platform:      Windows
*       From Screen:        BACKUP,     Record Number:    8
*       Variable:           mfile
*       Called By:          VALID Clause
*       Object Type:        List
*       Snippet Number:     5
*
*
*
FUNCTION _qsf0ko9lv     &&  mfile VALID
#REGION 1
mnewfile = filearray[mfile,1]
IF "[" $ mnewfile
   mnewpath = SUBSTR(mnewfile,2,LEN(mnewfile)-2)
   SET DEFA TO (mnewpath)
   DO newpathpop
   DO newfilepop
   SHOW GETS
ELSE
   mfname = mnewfile
   SHOW GETS
ENDIF

*
*
*       _QSFOKOAG9          mdrive WHEN
*
*       Function Origin:
*
*       From Platform:      MS-DOS
*       From Screen:        BACKUP,     Record Number:   14
*       Variable:           mdrive
*       Called By:          WHEN Clause
*       Object Type:        Popup
*       Snippet Number:     6
*
*
*
FUNCTION _qsf0koag9     &&  mdrive WHEN
#REGION 1
m.prevdrive = mdrive

*
*
*       _QSFOKOAID          mdrive VALID
*
*       Function Origin:
*
*       From Platform:      MS-DOS
*       From Screen:        BACKUP,     Record Number:   14
*       Variable:           mdrive
*       Called By:          VALID Clause
*       Object Type:        Popup
*       Snippet Number:     7
*
*
*
*Switch to the selected drive
FUNCTION _qsf0koaid     &&  mdrive VALID
#REGION 1
PRIVATE newdrive,mready

*Convert the popup bar number into the matching drive name
m.newdrive = drivearray[mdrive]

IF UPPER(m.newdrive) $ "A:B:"
   mready = yesno("Please insert disk into drive " + m.newdrive, "rea
=> dy","cancel")
ELSE
   mready = .T.
ENDIF
IF mready
   *Go there and reset all the other popups to match
   SET DEFAULT TO (m.newdrive)
   DO newpathpop
   DO newfilepop
ELSE
   mdrive =  m.prevdrive
   m.newdrive = drivearray[mdrive]
ENDIF
SHOW GETS

*
*
*       _QSFOKOANU          mpath VALID
*
*       Function Origin:
*
*       From Platform:      MS-DOS
*       From Screen:        BACKUP,     Record Number:   17
```

```
     *      │ Variable:             mpath
     *      │ Called By:            VALID Clause
     *      │ Object Type:          Popup
     *      │ Snippet Number:       8
     *      │
     *      └─────────────────────────────────────
     *
FUNCTION _qsf0koanu     &&  mpath VALID
#REGION 1
m.newdefault = pathstring()
SET DEFAULT TO (m.newdefault)
DO newpathpop
DO newfilepop
SHOW GETS

     *      ┌─────────────────────────────────────
     *      │
     *      │ _QSFOKOAQV            maction VALID
     *      │
     *      │ Function Origin:
     *      │
     *      │ From Platform:        MS-DOS
     *      │ From Screen:          BACKUP,      Record Number:   18
     *      │ Variable:             maction
     *      │ Called By:            VALID Clause
     *      │ Object Type:          Push Button
     *      │ Snippet Number:       9
     *      │
     *      └─────────────────────────────────────
     *
FUNCTION _qsf0koaqv     &&  maction VALID
#REGION 1
┌DO CASE
├CASE maction = 1
│
│   pkzip = .T.
│  ┌IF FILE(mfname)
│  │    mdel = yesno( mfname + " already exists, overwrite it? ", "Yes"
=> , "Cancel")
│  │  ┌IF mdel
│  │  │   DELETE FILE &mfname
│  │  ├ELSE
│  │  │   pkzip = .F.
│  │  └ENDIF
│  └ENDIF
│  ┌IF pkzip
│  │   DO pkzip
│  └ENDIF
├OTHERWISE
└ENDCASE

     *      ┌─────────────────────────────────────
     *      │
     *      │ _QSFOKOAUY            mfile VALID
     *      │
     *      │ Function Origin:
     *      │
     *      │ From Platform:        MS-DOS
     *      │ From Screen:          BACKUP,      Record Number:   19
     *      │ Variable:             mfile
     *      │ Called By:            VALID Clause
     *      │ Object Type:          List
     *      │ Snippet Number:       10
     *      │
     *      └─────────────────────────────────────
     *
FUNCTION _qsf0koauy     &&  mfile VALID
#REGION 1
mnewfile = filearray[mfile,1]
┌IF "[" $ mnewfile
│   mnewpath = SUBSTR(mnewfile,2,LEN(mnewfile)-2)
│   SET DEFA TO (mnewpath)
│   DO newpathpop
│   DO newfilepop
│   SHOW GETS
├ELSE
│   mfname = mnewfile
│   SHOW GETS
└ENDIF


     *      ┌─────────────────────────────────────
     *      │
     *      │  BACKUP/MS-DOS Supporting Procedures and Functions
     *      │
     *      │
     *      └─────────────────────────────────────

#REGION 1

     *      ┌─────────────────────────────────────
     *      │
     *      │          BACKUP Procedure NEWDRIVEPOP
     *      │
     *      │
     *      └─────────────────────────────────────


PROCEDURE newdrivepop
┌DO CASE
├CASE  DOS
│   ***********************
│   * Create a popup with each of the legal drive names
│   * The popup will be based on an array of the drive names
│   PRIVATE olddefault, olderror, drivename, ERROR
│
│   * We will change drives, so remember where we started
│   m.olddefault = SET( "DEFAULT" )
│
│   * To test for legal drives this program SET DEFAULT TO each drive
│   * If no error occurs the drive name will be added to an array
│
│   * Create the error trap
│   m.olderror = ON( "ERROR" )
│   m.error = .F.
│   ON ERROR m.error = .T.
│
│   * Create the array of legal drive names
│   DECLARE drivearray[ 1 ]
│   drivearray[ 1 ] = ""
│
│   * Loop from A to Z
│   m.drivename = "A"
│  ┌FOR i = 1 TO 26
│  │
│  │   * Try to switch to the drive
│  │   SET DEFAULT TO (m.drivename)
│  │
│  │   * If it worked
│  │  ┌IF NOT m.error
│  │  │
│  │  │   * Add the name to the last element in the array
```

```
            drivearray[ ALEN( DriveArray ) ] = m.drivename + ":"

            * Add another element at the end of the array
            DECLARE drivearray[ ALEN( DriveArray ) + 1 ]

         ENDIF

         * Change to the next letter in the alphabet
         m.drivename = CHR( ASC( m.drivename ) + 1 )

         * Reset the error trap
         m.error = .F.
      NEXT

      * If the first element is empty, no drives were found
      IF EMPTY( drivearray[ 1 ] )
         SHOW GET mdrive disabled
      ELSE
         * Drives were found so cut off the last empty element
         * added to the array in the loop
         DECLARE drivearray[ ALEN( DriveArray ) - 1 ]
      ENDIF

      * Reset the error trap and return to home
      ON ERROR &olderror
      SET DEFAULT TO (m.olddefault)

      * Initialize the popup variable to the element in the
      * drive array which contains the current drive
      mdrive = ASCAN( drivearray, m.olddefault )
<-----RETURN

      ********************
   CASE  WINDOWS
      ********************
      * Create a popup with each of the legal drive names
      * The popup will be based on an array of the drive names
      PRIVATE olddefault, olderror, drivename, ERROR

      * We will change drives, so remember where we started
      m.olddefault = SET( "DEFAULT" )

      * To test for legal drives this program SET DEFAULT TO each drive
      * If no error occurs the drive name will be added to an array

      * Create the error trap
      m.olderror = ON( "ERROR" )
      m.error = .F.
      ON ERROR m.error = .T.

      * Create the array of legal drive names
      DECLARE drivearray[ 3 ]
      drivearray[ 1 ] = "A"
      drivearray[ 2 ] = "B"
      drivearray[ 3 ] = ""

      * Loop from C to Z
      m.drivename = "C"
      FOR i = 3 TO 26

         * Try to switch to the drive
         SET DEFAULT TO (m.drivename)

         * If it worked
         IF NOT m.error

            * Add the name to the last element in the array
            drivearray[ ALEN( DriveArray ) ] = m.drivename + ":"

            * Add another element at the end of the array
            DECLARE drivearray[ ALEN( DriveArray ) + 1 ]

         ENDIF

         * Change to the next letter in the alphabet
         m.drivename = CHR( ASC( m.drivename ) + 1 )

         * Reset the error trap
         m.error = .F.
      NEXT

      * If the first element is empty, no drives were found
      IF EMPTY( drivearray[ 1 ] )
         SHOW GET mdrive disabled
      ELSE
         * Drives were found so cut off the last empty element
         * added to the array in the loop
         DECLARE drivearray[ ALEN( DriveArray ) - 1 ]
      ENDIF

      * Reset the error trap and return to home
      ON ERROR &olderror
      SET DEFAULT TO (m.olddefault)

      * Initialize the popup variable to the element in the
      * drive array which contains the current drive
      mdrive = ASCAN( drivearray, m.olddefault )
<-----RETURN

      ********************
   ENDCASE

   *
   *
   *                 BACKUP Procedure NEWPATHPOP
   *
   *
   *


   PROCEDURE newpathpop
   DO CASE
   CASE  DOS
      ********************
      * Create a popup with one bar for each subdirectory
      * in the current path
      PRIVATE dirstring, dircount

      * Base the popup on an array with one element for
      * each subdirectory in the current path
      *
      * Start the array with the current drive
      DECLARE patharray[ 1 ]
      patharray[ 1 ] = SET( "DEFAULT" )

      * Get the current path string
      m.dirstring = CURDIR()

      * If we are not in the root directory
      IF LEN( m.dirstring ) > 1

         * Start parsing the path string for each directory name
```

```
        m.dircount = 1

        * Continue as long as there is a pair of slashes
        * Surrounding the next part of the string
        DO WHILE AT( "\", m.dirstring, m.dircount ) <> 0 AND ;
              AT( "\", m.dirstring, m.dircount + 1 ) <> 0

           * Add another element at the end of the array
           DECLARE patharray[ m.DirCount + 1 ]

           * Cut out the next set of characters between the slashes
           m.firstchar = AT( "\", m.dirstring, m.dircount ) + 1

           m.pathlength = AT( "\", m.dirstring, m.dircount + 1 )  ;
              - m.firstchar

           * And add them to the next array element
           patharray[ m.DirCount + 1 ] = SUBSTR( m.dirstring, ;
              m.firstchar, m.pathlength )

           * Look for the next directory name in the path string
           m.dircount = m.dircount + 1

        ENDDO
     ENDIF

     * Point the popup variable at the last element in the array
     mpath = ALEN( patharray )
<----RETURN

     ********************
  CASE _WINDOWS
     ********************
     * Create a popup with one bar for each subdirectory
     * in the current path
     PRIVATE dirstring, dircount

     * Base the popup on an array with one element for
     * each subdirectory in the current path
     *
     * Start the array with the current drive
     DECLARE patharray[ 1 ]
     patharray[ 1 ] = SET( "DEFAULT" )

     * Get the current path string
     m.dirstring = CURDIR()

     * If we are not in the root directory
     IF LEN( m.dirstring ) > 1

        * Start parsing the path string for each directory name
        m.dircount = 1

        * Continue as long as there is a pair of slashes
        * Surrounding the next part of the string
        DO WHILE AT( "\", m.dirstring, m.dircount ) <> 0 AND ;
              AT( "\", m.dirstring, m.dircount + 1 ) <> 0

           * Add another element at the end of the array
           DECLARE patharray[ m.DirCount + 1 ]

           * Cut out the next set of characters between the slashes
           m.firstchar = AT( "\", m.dirstring, m.dircount ) + 1

           m.pathlength = AT( "\", m.dirstring, m.dircount + 1 )  ;
              - m.firstchar

           * And add them to the next array element
           patharray[ m.DirCount + 1 ] = SUBSTR( m.dirstring, ;
              m.firstchar, m.pathlength )

           * Look for the next directory name in the path string
           m.dircount = m.dircount + 1

        ENDDO
     ENDIF

     * Point the popup variable at the last element in the array
     mpath = ALEN( patharray )
<----RETURN

     ********************
ENDCASE

*
*
*                  BACKUP Procedure NEWFILEPOP
*
*
*


PROCEDURE newfilepop
DO CASE
CASE _DOS
     ********************

     * Create a popup with all of the file names and its subdirection i
=> n the current directory
     * This will be based on an array as well

     * Create an array with all the files matching the current wild car
=> d
     * ADIR() creates an array with 5 columns.

     PRIVATE startsort
     * Fill an array with directory names only
     =ADIR( dirarray, "", "D")
     SIZE = ALEN(dirarray,1)
     IF dirarray[ 1, 1 ] = "."
        =ADEL(dirarray,1)
        SIZE = SIZE - 1
        DECLARE dirarray[size,5]
     ELSE
        * We must be in the root directory "\"
        * Add one more row to the directory array
        SIZE = SIZE + 1
        DECLARE dirarray[ SIZE, 5 ]

        * Push all the rows down by one
        =AINS( dirarray, 1 )

        * Fill in the first directory name in the array
        * a bug in the popups makes it refuse to display a
        * prompt of "\", use "\\" to make one \ appear
        * even then the bar will automatically be non-selectable
        * which in this case is fine
        dirarray[ 1,1 ] = "\\"

        * Start the sort after the root name
     ENDIF

  IF ALEN( dirarray, 1 ) > 2

     * Sort the array starting at the starting row
     =ASORT( dirarray, AELEMENT( dirarray, 2, 1 ) )
  ENDIF
  FOR i = 1 TO ALEN(dirarray,1)
     dirarray[i,1] = "[" + dirarray[i,1] + "]"
  ENDFOR

  IF ADIR(filearray, m.wildcard) = 0
     SIZE = ALEN(dirarray,1)
     DECLARE filearray[size,5]
     =ACOPY(dirarray, filearray)
  ELSE
     stop = ALEN(dirarray,1)
     FOR i = 1 TO stop
        SIZE = ALEN(filearray,1)
        DECLARE filearray[size + 1,5]
        =AINS(filearray,1)
        filearray[1,1] = dirarray(alen[dirarray,1],1)
        IF ALEN(dirarray,1) > 1
           DECLARE dirarray[alen(dirarray,1)-1,5]
        ENDIF
     ENDFOR
  ENDIF
  mfile = 1
<=RETURN

  *******************
CASE  WINDOWS
  *******************

  * Create a popup with all of the file names and its subdirection i
=> n the current directory
  * This will be based on an array as well

  * Create an array with all the files matching the current wild car
=> d

  * ADIR() creates an array with 5 columns.

  PRIVATE startsort
  * Fill an array with directory names only
  =ADIR( dirarray, "", "D")
  SIZE = ALEN(dirarray,1)
  IF dirarray[ 1, 1 ] = "."
     =ADEL(dirarray,1)
     SIZE = SIZE - 1
     DECLARE dirarray[size,5]
  ELSE
     * We must be in the root directory "\"
     * Add one more row to the directory array
     SIZE = SIZE + 1
     DECLARE dirarray[ SIZE, 5 ]

     * Push all the rows down by one
     =AINS( dirarray, 1 )

     * Fill in the first directory name in the array
     * a bug in the popups makes it refuse to display a
     * prompt of "\", use "\\" to make one \ appear
     * even then the bar will automatically be non-selectable
     * which in this case is fine
     dirarray[ 1,1 ] = "\\"

     * Start the sort after the root name
  ENDIF

  IF ALEN( dirarray, 1 ) > 2

     * Sort the array starting at the starting row
     =ASORT( dirarray, AELEMENT( dirarray, 2, 1 ) )
  ENDIF
  FOR i = 1 TO ALEN(dirarray,1)
     dirarray[i,1] = "[" + dirarray[i,1] + "]"
  ENDFOR

  IF ADIR(filearray, m.wildcard) = 0
     SIZE = ALEN(dirarray,1)
     DECLARE filearray[size,5]
     =ACOPY(dirarray, filearray)
  ELSE
     stop = ALEN(dirarray,1)
     FOR i = 1 TO stop
        SIZE = ALEN(filearray,1)
        DECLARE filearray[size + 1,5]
        =AINS(filearray,1)
        filearray[1,1] = dirarray(alen[dirarray,1],1)
        IF ALEN(dirarray,1) > 1
           DECLARE dirarray[alen(dirarray,1)-1,5]
        ENDIF
     ENDFOR
  ENDIF
  mfile = 1
<=RETURN

  *******************
ENDCASE

*
*
*              BACKUP Function PATHSTRING
*
*
*


PROCEDURE pathstring
DO CASE
CASE  DOS
  ******************
  * Convert an array of subdirectories, such as PathArray,
  * into a legal path string for use with SET DEFAULT
  * Use the current value of the path popup as the
  * ending point on the path
  PRIVATE ppath

  * Start with the drive name
  m.ppath = patharray[ 1 ]

  * If the path popup is pointing to a subdirectory
  IF mpath > 1

     * Add all the subdirectories to the path string
     FOR i = 2 TO mpath
        m.ppath = m.ppath + "\" + patharray[ i ]

     NEXT
  ENDIF

  * End the path with one last slash for good luck
  m.ppath = m.ppath + "\"

```

```
<----RETURN m.ppath

      *******************
  CASE _WINDOWS
      *******************
      * Convert an array of subdirectories, such as PathArray,
      * into a legal path string for use with SET DEFAULT
      * Use the current value of the path popup as the
      * ending point on the path
      PRIVATE ppath

      * Start with the drive name
      m.ppath = patharray[ 1 ]

      * If the path popup is pointing to a subdirectory
      IF mpath > 1

         * Add all the subdirectories to the path string
         FOR i = 2 TO mpath
            m.ppath = m.ppath + "\" + patharray[ i ]

         NEXT
      ENDIF

      * End the path with one last slash for good luck
      m.ppath = m.ppath + "\"

<----RETURN m.ppath

      *******************
  ENDCASE

    *
    *
    *          BACKUP Procedure PKZIP
    *
    *
    *


    PROCEDURE pkzip
    DO CASE
    CASE _DOS
       *******************

       PRIVATE CURDIR, datadir, msel
       CURDIR = FULLPATH(CURDIR())
       IF !EMPTY(GETENV("CAMD"))
          datadir = FULLPATH(GETENV("CAMD"))
       ELSE
          datadir = CURDIR()
       ENDIF
       desdir = pathstring() + mfname
       SET DEFA TO (datadir)
       IF !EMPTY(GETENV("CAMDUTIL"))
          pkzipcom = "!" + GETENV("CAMDUTIL") + "\PKZIP &desdir *.dbf *.c
=> dx *.idx *.fpt"
       ELSE
          pkzipcom = "!PKZIP &desdir *.dbf *.cdx *.idx *.fpt"
       ENDIF
       &pkzipcom
       USE
       DELETE FILE t0000000.txt
       SET DEFA TO (CURDIR)
<----RETURN
    CASE _WINDOWS
       *******************

       PRIVATE CURDIR, datadir, msel
       CURDIR = FULLPATH(CURDIR())
       IF !EMPTY(GETENV("KBDATA"))
          datadir = FULLPATH(GETENV("KBDATA"))
       ELSE
          datadir = CURDIR()
       ENDIF
       desdir = pathstring() + mfname
       SET DEFA TO (datadir)
       IF !EMPTY(GETENV("CAMDUTIL"))
          pkzipcom = "!" + GETENV("CAMDUTIL") + "\PKZIP &desdir *.dbf *.c
=> dx *.idx *.fpt"
       ELSE
          pkzipcom = "!PKZIP &desdir *.dbf *.cdx *.idx *.fpt"
       ENDIF
       &pkzipcom
       USE
       DELETE FILE t0000000.txt
       SET DEFA TO (CURDIR)
<----RETURN
    ENDCASE
*:  EOF: BACKUP.ac1
```

```
*
*
*       08/09/94              KB.SPR                 09:38:51
*
*
*       Author's Name
*
*       Copyright (c) 1994 Company Name
*       Address
*       City,     Zip
*
*       Description:
*       This program was automatically generated by GENSCRN.
*
*
*

DO CASE
CASE _WINDOWS


   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
   ENDIF
   m.compstat = SET("COMPATIBLE")
   SET COMPATIBLE FOXPLUS

   m.rborder = SET("READBORDER")
   SET readborder ON

   *
   *
   *            Windows Window definitions
   *
   *
   *

   IF NOT WEXIST("w_kb") ;
         OR UPPER(WTITLE("W_KB")) == "W_KB.PJX" ;
         OR UPPER(WTITLE("W_KB")) == "W_KB.SCX" ;
         OR UPPER(WTITLE("W_KB")) == "W_KB.MNX" ;
         OR UPPER(WTITLE("W_KB")) == "W_KB.PRG" ;
         OR UPPER(WTITLE("W_KB")) == "W_KB.FRX" ;
         OR UPPER(WTITLE("W_KB")) == "W_KB.QPR"
      DEFINE WINDOW w_kb ;
         AT  0.000, 0.000 ;
         SIZE 30.167,69.250 ;
         TITLE "Knowledge Base Area" ;
         FONT "Terminal", 8 ;
         FLOAT ;
         CLOSE ;
         SHADOW ;
         NOMINIMIZE
      MOVE WINDOW w_kb CENTER
   ENDIF


   *
   *
   *            KB/Windows Setup Code - SECTION 2
   *
   *
   *

   #REGION 1
   PRIVATE areaname, areaid
   PRIVATE mquals, mgoals, mdispl
   DIMENSION mquals[10,4],mgoals[10,4],mdispl[10,4]
   DIMENSION methods[4]
   DIMENSION infs[2]
   methods[1] = "BAYES"
   methods[2] = "ADDITIVE"
   methods[3] = "FUZZY"
   methods[4] = "NONE"
   infs[1]    = "FORWARD"
   infs[2]    = "BACKWARD"

   SELECT 0
   USE area INDEX area
   SELECT 0
   USE disease INDEX disid, disease
   SELECT 0
   USE enum INDEX enum
   SELECT 0
   USE VAL INDEX VAL
   SELECT 0
   USE dict INDEX dictid, dictname
   SET RELATION TO id INTO enum, id INTO VAL
   SELECT 0
   USE goals INDEX goals
   SET FILTER TO area = area.area
   SELECT 0
   USE DISPLAY
   SET FILTER TO area = area.area
   SELECT 0
   USE quals INDEX quals
   SET FILTER TO area = area.area
   SELECT area
   SCATTER MEMVAR
   m.areaname = m.name
   m.areaid = m.area
   DO setquals
   DO setgoals
   DO setdisplay
   SELECT area
   m.mpop = area.method + 1
   m.mpop2 = area.inference + 1


   *
   *
   *            KB/Windows Screen Layout
   *
   *
   *

   #REGION 1
   IF WVISIBLE("w_kb")
      ACTIVATE WINDOW w_kb SAME
   ELSE
      ACTIVATE WINDOW w_kb NOSHOW
   ENDIF
   @ 9.250,4.500 SAY "Consider"  ;
      FONT "Terminal", 8
   @ 14.000,4.750 SAY "Probable"  ;
      FONT "Terminal", 8
   @ 18.417,5.000 SAY "Likely"  ;
      FONT "Terminal", 8
   @ 7.333,3.250 TO 21.666,24.875 ;
      PEN 1, 8
   @ 5.833,4.000 SAY "Display thresholds"  ;
      FONT "Terminal", 8
   @ 15.917,26.250 SAY "Inference"  ;
      FONT "Terminal", 8
   @ 20.583,26.000 SAY "Confidence"  ;
      FONT "Terminal", 8
   @ 1.500,4.125 SAY "Knowledge Base Area:"  ;
      FONT "Terminal", 8 ;
      STYLE "T"
   @ 8.500,26.375 SAY "Total Rules:"  ;
      FONT "Terminal", 8 ;
      STYLE "T"
   @ 25.083,5.625 GET mbuttons ;
      PICTURE "@*HN \<Add;\<Edit;\<Delete;\<OK;\<Cancel" ;
      SIZE 1.917,11.125,0.875 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kofx9()
   @ 6.083,53.500 GET mbbuttons ;
      PICTURE "@*VN \<Next;\<Previous;\<Browse;\<Qualifiers;\<Goals;Conc\<Lusion" ;
      SIZE 2.000,11.625,1.083 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kog2m()
   @ 1.583,24.875 GET m.areaname ;
      SIZE 1.000,34.750 ;
      DEFAULT " " ;
      FONT "Terminal", 8
   @ 9.250,13.625 GET m.threshold ;
      SIZE 1.333,8.750 ;
      DEFAULT 0 ;
      FONT "Terminal", 8
   @ 14.000,13.875 GET m.probable ;
      SIZE 1.333,9.000 ;
      DEFAULT 0 ;
      FONT "Terminal", 8
   @ 18.500,13.875 GET m.likely ;
      SIZE 1.417,9.250 ;
      DEFAULT 0 ;
      FONT "Terminal", 8
   @ 8.500,39.750 GET m.rules ;
      SIZE 1.000,5.875 ;
      DEFAULT 0 ;
      FONT "Terminal", 8 ;
      DISABLE
   @ 11.917,26.500 GET m.isdiag ;
      PICTURE "@*C DIAGNOSIS" ;
      SIZE 1.417,11.875 ;
      DEFAULT 0 ;
      FONT "Terminal", 8
   @ 20.250,37.500 GET m.mpop ;
      PICTURE "@^" ;
      FROM methods ;
      SIZE 1.500,13.000 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kogfv()
   @ 15.833,37.500 GET m.mpop2 ;
      PICTURE "@^" ;
      FROM infs ;
      SIZE 1.500,12.750 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kogih()

   IF NOT WVISIBLE("w_kb")
      ACTIVATE WINDOW w_kb
   ENDIF

   READ CYCLE MODAL

   RELEASE WINDOW w_kb

   #REGION 0

   SET readborder &rborder

   IF m.talkstat = "ON"
      SET TALK ON
   ENDIF
   IF m.compstat = "ON"
      SET COMPATIBLE ON
   ENDIF


   *
   *
   *               KB/Windows Cleanup Code
   *
   *
   *

   #REGION 1
   SELECT area
   USE
   SELECT goals
   USE
   SELECT DISPLAY
   USE
   SELECT quals
   USE
   SELECT disease
   USE
   SELECT enum
   USE
   SELECT dict
   USE
```

```
    SELECT VAL
    USE
    RETURN

CASE _DOS

    #REGION 0
    REGIONAL m.currarea, m.talkstat, m.compstat

    IF SET("TALK") = "ON"
       SET TALK OFF
       m.talkstat = "ON"
    ELSE
       m.talkstat = "OFF"
    ENDIF
    m.compstat = SET("COMPATIBLE")
    SET COMPATIBLE FOXPLUS

    *
    *
    *          MS-DOS Window definitions
    *
    *
    *

    IF NOT WEXIST("w_kb") ;
          OR UPPER(WTITLE("W_KB")) == "W_KB.PJX" ;
          OR UPPER(WTITLE("W_KB")) == "W_KB.SCX" ;
          OR UPPER(WTITLE("W_KB")) == "W_KB.MNX" ;
          OR UPPER(WTITLE("W_KB")) == "W_KB.PRG" ;
          OR UPPER(WTITLE("W_KB")) == "W_KB.FRX" ;
          OR UPPER(WTITLE("W_KB")) == "W_KB.QPR"
       DEFINE WINDOW w_kb ;
          FROM INT((SROW()-18)/2),INT((SCOL()-75)/2) ;
          TO INT((SROW()-18)/2)+17,INT((SCOL()-75)/2)+74 ;
          TITLE "Knowledge Base Area" ;
          FLOAT ;
          CLOSE ;
          SHADOW ;
          NOMINIMIZE ;
          COLOR SCHEME 1
    ENDIF

    *
    *
    *          KB/MS-DOS Setup Code - SECTION 2
    *
    *
    *

    #REGION 1
    SCATTER MEMVAR
    m.mpop = m.method + 1
    m.mpop2 = m.inference + 1

    *
    *
    *          KB/MS-DOS Screen Layout
    *
    *
    *

    #REGION 1
    IF WVISIBLE("w_kb")
       ACTIVATE WINDOW w_kb SAME
    ELSE
       ACTIVATE WINDOW w_kb NOSHOW
    ENDIF
    @ 5,2 SAY "Consider" ;
       SIZE 1,8, 0
    @ 7,2 SAY "Probable" ;
       SIZE 1,8, 0
    @ 9,2 SAY "Likely" ;
       SIZE 1,6, 0
    @ 14,6 GET mbuttons ;
       PICTURE "@*HT \<Add;\<Edit;\<Delete;\<OK;\<Cancel" ;
       SIZE 1,12,1 ;
       DEFAULT 1 ;
       VALID _qsf0koh4z()
    @ 5,12 GET m.threshold ;
       SIZE 1,6 ;
       DEFAULT 0
    @ 7,12 GET m.probable ;
       SIZE 1,6 ;
       DEFAULT 0
    @ 9,12 GET m.likely ;
       SIZE 1,6 ;
       DEFAULT 0
    @ 1,17 GET m.name ;
       SIZE 1,34 ;
       DEFAULT " "
    @ 3,0 TO 10,24
    @ 3,4 SAY "Display thresholds" ;
       SIZE 1,18, 0
    @ 3,28 SAY "Rules" ;
       SIZE 1,5, 0
    @ 3,37 GET m.rules ;
       SIZE 1,4 ;
       DEFAULT " " ;
       DISABLE
    @ 9,39 GET m.mpop ;
       PICTURE "@^" ;
       FROM methods ;
       SIZE 3,12 ;
       DEFAULT 1 ;
       VALID _qsf0kohj1() ;
       COLOR SCHEME 1, 2
    @ 7,28 SAY "Inference" ;
       SIZE 1,9, 0
    @ 6,39 GET m.mpop2 ;
       PICTURE "@^" ;
       FROM infs ;
       SIZE 3,12 ;
```

```
            DEFAULT 1 ;
            VALID _qsf0kohm5() ;
            COLOR SCHEME 1, 2
        @ 10,28 SAY "Confidence" ;
            SIZE 1,10, 0
        @ 5,28 GET m.isdiag ;
            PICTURE "@*C DIAGNOSIS" ;
            SIZE 1,13 ;
            DEFAULT 0
        @ 1,0 GET minvbutton ;
            PICTURE "@*HN \<Knowledge Base" ;
            SIZE 1,16,1 ;
            DEFAULT 1 ;
            VALID _qsf0kohpt() ;
            MESSAGE "Knowledge Base"
        @ 2,58 GET mbbuttons ;
            PICTURE "@*VN \<Next;\<Previous;\<Browse;\<Qualifiers;\<Goals;C
=> onc\<Lusion" ;
            SIZE 1,13,1 ;
            DEFAULT 1

        IF NOT WVISIBLE("w_kb")
            ACTIVATE WINDOW w_kb
        ENDIF

        READ CYCLE MODAL

        RELEASE WINDOW w_kb

        #REGION 0
        IF m.talkstat = "ON"
            SET TALK ON
        ENDIF
        IF m.compstat = "ON"
            SET COMPATIBLE ON
        ENDIF


    ENDCASE


      *
      *
      *        _QSFOKOFX9          mbuttons VALID
      *
      *        Function Origin:
      *
      *        From Platform:      Windows
      *        From Screen:        KB,      Record Number:   11
      *        Variable:           mbuttons
      *        Called By:          VALID Clause
      *        Object Type:        Push Button
      *        Snippet Number:     1
      *
      *
      *
      FUNCTION _qsf0kofx9     &&  mbuttons VALID
      #REGION 1
      DO CASE
      CASE mbuttons = 1           && <Add>
         SET TOPIC TO "ADD"
         DO kbload.spr
      CASE mbuttons = 2                   && <Edit>
         SET TOPIC TO "kbedit"
         DO kbedit.spr
      CASE mbuttons = 3           && <Delete>
         SET TOPIC TO "DELETE"
         DO kbdel.spr
      CASE mbuttons = 4                && <ok>
         SELECT area
         m.name = m.areaname
         GATHER MEMVAR
         CLEAR READ
      CASE mbuttons = 5                && <cancel>
         CLEAR READ
      ENDCASE
      SELECT area
      SCATTER MEMVAR
      DO setquals
      DO setgoals
      DO setdisplay
      m.mpop = area.method + 1
      m.mpop2 = area.inference + 1
      SHOW GETS
      mtopic = ALIAS()
      SET TOPIC TO &mtopic


      *
      *
      *        _QSFOKOG2M          mbbuttons VALID
      *
      *        Function Origin:
      *
      *        From Platform:      Windows
      *        From Screen:        KB,      Record Number:   12
      *        Variable:           mbbuttons
      *        Called By:          VALID Clause
      *        Object Type:        Push Button
      *        Snippet Number:     2
      *
      *
      *
      FUNCTION _qsf0kog2m     &&  mbbuttons VALID
      #REGION 1
      SELECT area
      DO CASE
      CASE mbbuttons = 1           && <Next>
         m.recno = RECNO()
         IF !EOF()
            SKIP
            SHOW GET mbbuttons,2 ENABLE
            IF EOF()
               GOTO BOTTOM
               SHOW GET mbbuttons,1 DISABLE
            ENDIF
         ELSE
            GOTO BOTTOM
            SHOW GET mbbuttons,2 ENABLE
            SHOW GET mbbuttons,1 DISABLE
         ENDIF
      CASE mbbuttons = 2           && <previous>
         m.recno = RECNO()
         IF !BOF()
            SKIP -1
            SHOW GET mbbuttons,1 ENABLE
         ELSE
            GOTO TOP
            SHOW GET mbbuttons,1 ENABLE
            SHOW GET mbbuttons,2 DISABLE
         ENDIF
      CASE mbbuttons = 3           && <Browse>
```

```
        SET TOPIC TO "BROWSE"
        BROWSE NOEDIT
    CASE mbbuttons = 4          && <Qualifiers>
        PRIVATE msel
        msel = SELECT()
        SELECT quals
        SET TOPIC TO "QUALIFIERS"
        DO qual.spr
        SELECT (msel)
 <----  RETURN .T.
    CASE mbbuttons = 5          && <Goals>
        PRIVATE msel
        msel = SELECT()
        SELECT goals
        SET TOPIC TO "GOALS"
        DO goal.spr
        SELECT (msel)
 <----  RETURN .T.
    CASE mbbuttons = 6          && <conclusion>
        PRIVATE msel
        msel = SELECT()
        SELECT DISPLAY
        SET TOPIC TO "DISPLAY"
        DO display.spr
        SELEC (msel)
 <----  RETURN .T.
    ENDCASE
    SCATTER MEMVAR
    m.areaname = m.name
    m.areaid = m.area
    DO setquals
    DO setgoals
    DO setdisplay
    m.mpop = area.method + 1
    m.mpop2 = area.inference + 1
    SHOW GETS
    mtopic = ALIAS()
    SET TOPIC TO &mtopic


    *
    *
    *        _QSFOKOGFV           m.mpop VALID
    *
    *        Function Origin:
    *
    *        From Platform:       Windows
    *        From Screen:         KB,     Record Number:   19
    *        Variable:            m.mpop
    *        Called By:           VALID Clause
    *        Object Type:         Popup
    *        Snippet Number:      3
    *
    *
    *
    FUNCTION _qsf0kogfv     &&  m.mpop VALID
    #REGION 1
    m.method = m.mpop - 1


    *
    *
    *        _QSFOKOGIH           m.mpop2 VALID
    *
    *        Function Origin:
    *
    *        From Platform:       Windows
    *        From Screen:         KB,     Record Number:   20
    *        Variable:            m.mpop2
    *        Called By:           VALID Clause
    *        Object Type:         Popup
    *        Snippet Number:      4
    *
    *
    *
    FUNCTION _qsf0kogih     &&  m.mpop2 VALID
    #REGION 1
    m.inference = m.mpop2 - 1


    *
    *
    *        _QSFOKOH4Z           mbuttons VALID
    *
    *        Function Origin:
    *
    *        From Platform:       MS-DOS
    *        From Screen:         KB,     Record Number:   26
    *        Variable:            mbuttons
    *        Called By:           VALID Clause
    *        Object Type:         Push Button
    *        Snippet Number:      5
    *
    *
    *
    FUNCTION _qsf0koh4z     &&  mbuttons VALID
    #REGION 1
    DO CASE
    CASE mbuttons = 1           && <Add>
        SET TOPIC TO "ADD"
        DO kbload.spr
    CASE mbuttons = 2           && <Delete>
        SET TOPIC TO "DELETE"
        DO kbdel.spr
    CASE mbuttons = 3           && <conclusion>
        SELECT DISPLAY
        SET TOPIC TO "DISPLAY"
        DO display.spr
 <----  RETURN .T.
    CASE mbuttons = 4           && ok
        GATHER MEMVAR
    CASE mbuttons = 5           && cancel
        SCATTER MEMVAR
    ENDCASE

    DO CASE
    CASE mbbutton = 1 .OR. mbbutton = 2      && <Next> <Previous>
        IF ALIAS() != "AREA" .AND. ALIAS() != "RULE"
 <----      RETURN .T.
        ENDIF
        m.recno = RECNO()
        IF mbbutton = 1  .AND. !EOF()        && <Next>
            SKIP
        ENDIF
        IF mbbutton = 2  .AND.  !BOF()       && <Previous>
            SKIP -1
        ENDIF
        DO CASE
        CASE ALIAS() = "RULE"
            m.goback  = area != area.area
        OTHERWISE
            m.goback = EOF()
```

```
     ENDCASE
     IF m.goback
        GOTO m.recno
     ENDIF
  CASE mbbutton = 3                        && <Browse>
     SET TOPIC TO "BROWSE"
     DO CASE
     CASE ALIAS() = "AREA"
        BROWSE NOEDIT
     CASE ALIAS() = "RULE"
        BROWSE FOR area = area.area NOEDIT
     ENDCASE
  CASE mbbutton = 4                        && <Edit>
     SET TOPIC TO "EDIT"
     DO CASE
     CASE ALIAS() = "AREA"
        DO kb.spr
     CASE ALIAS() = "RULE"
        DO rule.spr
     ENDCASE
  CASE mbbutton = 5
     SELECT quals
     SET TOPIC TO "QUALIFIERS"
     DO qual.spr
     RETURN .T.
  CASE mbbutton = 6
     SELECT goals
     SET TOPIC TO "GOALS"
     DO goal.spr
     RETURN .T.
  ENDCASE
  DO setprem
  DO setact
  DO setquals
  DO setgoals
  DO setdisplay
  m.mpop = area.method + 1
  m.mpop2 = area.inference + 1
  SHOW GETS
  mtopic = ALIAS()
  SET TOPIC TO &mtopic


  *
  *
  *        _QSFOKOHJ1          m.mpop VALID
  *
  *        Function Origin:
  *
  *        From Platform:      MS-DOS
  *        From Screen:        KB,      Record Number:   35
  *        Variable:           m.mpop
  *        Called By:          VALID Clause
  *        Object Type:        Popup
  *        Snippet Number:     6
  *
  *
  *
  FUNCTION _qsf0kohj1      &&  m.mpop VALID
  #REGION 1
  m.method = m.mpop - 1


  *
  *
  *        _QSFOKOHM5          m.mpop2 VALID
  *
  *        Function Origin:
  *
  *        From Platform:      MS-DOS
  *        From Screen:        KB,      Record Number:   37
  *        Variable:           m.mpop2
  *        Called By:          VALID Clause
  *        Object Type:        Popup
  *        Snippet Number:     7
  *
  *
  *
  FUNCTION _qsf0kohm5      &&  m.mpop2 VALID
  #REGION 1
  m.inference = m.mpop2 - 1


  *
  *
  *        _QSFOKOHPT          minvbutton VALID
  *
  *        Function Origin:
  *
  *        From Platform:      MS-DOS
  *        From Screen:        KB,      Record Number:   40
  *        Variable:           minvbutton
  *        Called By:          VALID Clause
  *        Object Type:        Push Button
  *        Snippet Number:     8
  *
  *
  *
  FUNCTION _qsf0kohpt      &&  minvbutton VALID
  #REGION 1
  SELECT area
  SET TOPIC TO "AREA"



  *
  *
  *        KB/MS-DOS Supporting Procedures and Functions
  *
  *
  *

  #REGION 1

  *
  *
  *                     KB Procedure SETQUALS
  *
  *
  *

  PROCEDURE setquals
  PRIVATE msel,i
  msel = SELECT()
  SELECT quals
  nq = RECCOUNT()
  nq = MAX(nq,1)
  DIMENSION mquals[nq,4]
  GOTO TOP
  mquals = ""
  i =0
  DO WHILE !EOF()
```

```
        i = i + 1
        mquals[i,3]  = OBJECT
        mquals[i,4]  = id
        IF OBJECT = "D"
           SELECT disease
        ELSE
           SELECT dict
        ENDIF
        SEEK mquals[i,4]
        mquals[i,1] = name
        mquals[i,2] = ALIAS
        SELECT quals
        SKIP
     ENDDO
     nq = i
     SELECT (msel)
<----RETURN


     *
     *
     *             KB Procedure SETGOALS
     *
     *
     *

     PROCEDURE setgoals
     PRIVATE msel, i
     msel = SELECT()
     SELECT goals
     ng = RECCOUNT()
     ng = MAX(ng,1)
     DIMENSION mgoals[ng,4]
     GOTO TOP
     i = 0
     mgoals = ""
     DO WHILE !EOF()
        i = i + 1
        mgoals[i,3]  = OBJECT
        mgoals[i,4]  = id
        IF OBJECT = "D"
           SELECT disease
        ELSE
           SELECT dict
        ENDIF
        SEEK mgoals[i,4]
        mgoals[i,1] = name
        mgoals[i,2] = ALIAS
        SELECT goals
        SKIP
     ENDDO
     ng = i
     SELECT (msel)
<----RETURN


     *
     *
     *             KB Procedure SETDISPLAY
     *
     *
     *

     PROCEDURE setdisplay
     PRIVATE msel, i
     msel = SELECT()
     SELECT DISPLAY
     nd = RECCOUNT()
     nd = MAX(nd,1)
     DIMENSION mdispl[nd,4]
     GOTO TOP
     i = 0
     mdispl = ""
     DO WHILE !EOF()
        i = i + 1
        mdispl[i,3]  = OBJECT
        mdispl[i,4]  = id
        IF OBJECT = "D"
           SELECT disease
        ELSE
           SELECT dict
        ENDIF
        SEEK mdispl[i,4]
        mdispl[i,1] = name
        mdispl[i,2] = ALIAS
        SELECT DISPLAY
        SKIP
     ENDDO
     nd = i
     SELECT (msel)
<----RETURN


     *
     *
     *             KB Procedure EDGOAL
     *
     *
     *

     PROCEDURE edgoal
     =edobj( mgoals[mg,3], mgoals[mg,4] )
<----RETURN


     *
     *
     *             KB Procedure EDQUAL
     *
     *
     *

     PROCEDURE edqual
     =edobj( mquals[mq,3], mquals[mq,4] )
<----RETURN


     *
     *
     *             KB Procedure EDDISPLAY
     *
     *
     *

     PROCEDURE eddisplay
     =edobj( mdispl[md,3], mdispl[md,4] )
<----RETURN


     *
     *
     *             KB Function EDOBJ
```

```
*
*
*

FUNCTION edobj
PARAMETERS mobj, mid
PRIVATE msel
msel = SELECT()
┌IF mobj = "D"
│   SELECT disease
│   SEEK mid
│   DO disease.spr
├ELSE
│   SELECT dict
│   SEEK mid
│   DO dict.spr WITH mid
└ENDIF
 SELECT (msel)
<──RETURN ""


*:  EOF: KB.ac1
```

```
*
*
*       08/09/94              KBDEL.SPR              09:38:58
*
*
*
*       Author's Name
*
*       Copyright (c) 1994 Company Name
*       Address
*       City,     Zip
*
*       Description:
*       This program was automatically generated by GENSCRN.
*
*

DO CASE
CASE _WINDOWS

   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
   ENDIF
   m.compstat = SET("COMPATIBLE")
   SET COMPATIBLE FOXPLUS

   m.rborder = SET("READBORDER")
   SET readborder ON

   *
   *
   *              Windows Window definitions
   *
   *
   *

   IF NOT WEXIST("_qsf0kok6q")
      DEFINE WINDOW _qsf0kok6q ;
         AT  0.000, 0.000  ;
         SIZE 7.917,42.500 ;
         TITLE "Knowledge Base Delete" ;
         FONT "Terminal", 8 ;
         FLOAT ;
         NOCLOSE ;
         SHADOW ;
         NOMINIMIZE ;
         DOUBLE
      MOVE WINDOW _qsf0kok6q CENTER
   ENDIF

   *
   *
   *              KBDEL/Windows Screen Layout
   *
   *
   *

   #REGION 1
   IF WVISIBLE("_qsf0kok6q")
      ACTIVATE WINDOW _qsf0kok6q SAME
   ELSE
      ACTIVATE WINDOW _qsf0kok6q NOSHOW
   ENDIF
   @ 4.250,10.375 GET mbutton ;
      PICTURE "@*HT OK;Cancel" ;
      SIZE 2.000,8.250,1.500 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0okeu()
   @ 1.333,9.125 GET area.name ;
      SIZE 1.000,31.250 ;
      DEFAULT " " ;
      FONT "Terminal", 8 ;
      DISABLE
   @ 1.250,0.750 SAY "Delete:"  ;
      FONT "Terminal", 8

   IF NOT WVISIBLE("_qsf0kok6q")
      ACTIVATE WINDOW _qsf0kok6q
   ENDIF

   READ CYCLE MODAL

   RELEASE WINDOW _qsf0kok6q

   #REGION 0

   SET readborder &rborder

   IF m.talkstat = "ON"
      SET TALK ON
   ENDIF
   IF m.compstat = "ON"
      SET COMPATIBLE ON
   ENDIF


CASE _DOS


   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
   ENDIF
   m.compstat = SET("COMPATIBLE")
   SET COMPATIBLE FOXPLUS

   *
```

```
*
*                         MS-DOS Window definitions
*
*
*

IF NOT WEXIST("_qsf0kokln")
   DEFINE WINDOW _qsf0kokln ;
      FROM INT((SROW()-8)/2),INT((SCOL()-39)/2) ;
      TO INT((SROW()-8)/2)+7,INT((SCOL()-39)/2)+38 ;
      TITLE "Knowledge Base Delete" ;
      FLOAT ;
      NOCLOSE ;
      SHADOW ;
      NOMINIMIZE ;
      DOUBLE ;
      COLOR SCHEME 5
ENDIF


*
*
*                         KBDEL/MS-DOS Screen Layout
*
*
*

#REGION 1
IF WVISIBLE("_qsf0kokln")
   ACTIVATE WINDOW _qsf0kokln SAME
ELSE
   ACTIVATE WINDOW _qsf0kokln NOSHOW
ENDIF
@ 4,8 GET mbutton ;
   PICTURE "@*HT OK;Cancel" ;
   SIZE 1,8,2 ;
   DEFAULT 1 ;
   VALID _qsf0kokpt()
@ 1,10 GET area.name ;
   SIZE 1,26 ;
   DEFAULT " " ;
   DISABLE
@ 1,1 SAY "Delete:" ;
   SIZE 1,7, 0

IF NOT WVISIBLE("_qsf0kokln")
   ACTIVATE WINDOW _qsf0kokln
ENDIF

READ CYCLE MODAL

RELEASE WINDOW _qsf0kokln

#REGION 0
IF m.talkstat = "ON"
   SET TALK ON
ENDIF
   IF m.compstat = "ON"
      SET COMPATIBLE ON
   ENDIF


ENDCASE


*
*
*     _QSF0KOKEU          mbutton VALID
*
*     Function Origin:
*
*     From Platform:      Windows
*     From Screen:        KBDEL,     Record Number:    2
*     Variable:           mbutton
*     Called By:          VALID Clause
*     Object Type:        Push Button
*     Snippet Number:     1
*
*
*
FUNCTION _qsf0kokeu     &&  mbutton VALID
#REGION 1
DO CASE
CASE mbutton = 1
   *do (m.home + "\kbdelete") with area.name
   DO kbdelete WITH area.name
ENDCASE
RETURN .T.

*
*
*     _QSF0KOKPT          mbutton VALID
*
*     Function Origin:
*
*     From Platform:      MS-DOS
*     From Screen:        KBDEL,     Record Number:    7
*     Variable:           mbutton
*     Called By:          VALID Clause
*     Object Type:        Push Button
*     Snippet Number:     2
*
*
*
FUNCTION _qsf0kokpt     &&  mbutton VALID
#REGION 1
DO CASE
CASE mbutton = 1
   *do (m.home + "\kbdelete") with area.name
   DO kbdelete WITH area.name
ENDCASE
RETURN .T.
*:  EOF: KBDEL.ac1
```

```
*
*       08/09/94              QUAL.SPR              09:39:00
*
*
*       Author's Name
*
*       Copyright (c) 1994 Company Name
*       Address
*       City,     Zip
*
*       Description:
*       This program was automatically generated by GENSCRN.
*
*
*

DO CASE
CASE _WINDOWS


   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
   ENDIF
   m.compstat = SET("COMPATIBLE")
   SET COMPATIBLE FOXPLUS

   m.rborder = SET("READBORDER")
   SET readborder ON

   *
   *
   *              Windows Window definitions
   *
   *
   *

   IF NOT WEXIST("w_qe") ;
         OR UPPER(WTITLE("W_QE")) == "W_QE.PJX" ;
         OR UPPER(WTITLE("W_QE")) == "W_QE.SCX" ;
         OR UPPER(WTITLE("W_QE")) == "W_QE.MNX" ;
         OR UPPER(WTITLE("W_QE")) == "W_QE.PRG" ;
         OR UPPER(WTITLE("W_QE")) == "W_QE.FRX" ;
         OR UPPER(WTITLE("W_QE")) == "W_QE.QPR"
      DEFINE WINDOW w_qe ;
         AT  0.000, 0.000  ;
         SIZE 19.923,57.333 ;
         TITLE "Qualifier Editor" ;
         FONT "MS Sans Serif", 8 ;
         STYLE "B" ;
         FLOAT ;
         CLOSE ;
         SHADOW ;
         NOMINIMIZE ;
         COLOR RGB(,,,128,128,128)
      MOVE WINDOW w_qe CENTER
   ENDIF



   *
   *
   *              QUAL/Windows Setup Code - SECTION 2
   *
   *
   *

   #REGION 1
   EXTERNAL ARRAY mquals

   *
   *
   *              QUAL/Windows Screen Layout
   *
   *
   *

   #REGION 1
   IF WVISIBLE("w_qe")
      ACTIVATE WINDOW w_qe SAME
   ELSE
      ACTIVATE WINDOW w_qe NOSHOW
   ENDIF
   @ 17.538,16.333 GET mbuttons ;
      PICTURE "@*HT \<Edit;\<Quit" ;
      SIZE 1.846,10.667,1.000 ;
      DEFAULT 1 ;
      FONT "MS Sans Serif", 8 ;
      STYLE "B" ;
      VALID _qsf0komjf()
   @ 0.385,2.667 GET mq ;
      PICTURE "@&N" ;
      FROM mquals ;
      SIZE 16.154,63.200 ;
      DEFAULT 1 ;
      FONT "MS Sans Serif", 8 ;
      VALID _qsf0komrx()

   IF NOT WVISIBLE("w_qe")
      ACTIVATE WINDOW w_qe
   ENDIF

   READ CYCLE MODAL

   RELEASE WINDOW w_qe

   #REGION 0

   SET readborder &rborder

   IF m.talkstat = "ON"
      SET TALK ON
```

```
    └ENDIF
    ┌IF m.compstat = "ON"
    │   SET COMPATIBLE ON
    └ENDIF


      *    ┌──────────────────────────────────────────
      *    │
      *    │         QUAL/Windows Cleanup Code
      *    │
      *    └──────────────────────────────────────────
      *

      #REGION 1
 <────RETURN

  ┌CASE _DOS


      #REGION 0
      REGIONAL m.currarea, m.talkstat, m.compstat

     ┌IF SET("TALK") = "ON"
     │   SET TALK OFF
     │   m.talkstat = "ON"
     ├ELSE
     │   m.talkstat = "OFF"
     └ENDIF
      m.compstat = SET("COMPATIBLE")
      SET COMPATIBLE FOXPLUS

      *    ┌──────────────────────────────────────────
      *    │
      *    │         MS-DOS Window definitions
      *    │
      *    └──────────────────────────────────────────
      *

     ┌IF NOT WEXIST("w_qe") ;
     │       OR UPPER(WTITLE("W_QE")) == "W_QE.PJX" ;
     │       OR UPPER(WTITLE("W_QE")) == "W_QE.SCX" ;
     │       OR UPPER(WTITLE("W_QE")) == "W_QE.MNX" ;
     │       OR UPPER(WTITLE("W_QE")) == "W_QE.PRG" ;
     │       OR UPPER(WTITLE("W_QE")) == "W_QE.FRX" ;
     │       OR UPPER(WTITLE("W_QE")) == "W_QE.QPR"
     │   DEFINE WINDOW w_qe ;
     │       FROM INT((SROW()-17)/2),INT((SCOL()-58)/2) ;
     │       TO INT((SROW()-17)/2)+16,INT((SCOL()-58)/2)+57 ;
     │       TITLE "Qualifier Editor" ;
     │       FLOAT ;
     │       CLOSE ;
     │       SHADOW ;
     │       NOMINIMIZE ;
     │       COLOR SCHEME 1
     └ENDIF


      *    ┌──────────────────────────────────────────
      *    │
      *    │    QUAL/MS-DOS Setup Code - SECTION 2
      *    │
      *    └──────────────────────────────────────────
      *

      #REGION 1
      EXTERNAL ARRAY mquals

      *    ┌──────────────────────────────────────────
      *    │
      *    │       QUAL/MS-DOS Screen Layout
      *    │
      *    └──────────────────────────────────────────
      *

      #REGION 1
     ┌IF WVISIBLE("w_qe")
     │   ACTIVATE WINDOW w_qe SAME
     ├ELSE
     │   ACTIVATE WINDOW w_qe NOSHOW
     └ENDIF
      @ 14,11 GET mbuttons ;
         PICTURE "@*HN \<Add;\<Delete;\<OK;\<Cancel" ;
         SIZE 1,8,1 ;
         DEFAULT 1 ;
         VALID _qsf0kon7e()
      @ 0,1 GET mq ;
         PICTURE "@&N" ;
         FROM mquals ;
         SIZE 13,54 ;
         DEFAULT 1 ;
         VALID _qsf0konbu() ;
         COLOR SCHEME 2

     ┌IF NOT WVISIBLE("w_qe")
     │   ACTIVATE WINDOW w_qe
     └ENDIF

      READ CYCLE MODAL

      RELEASE WINDOW w_qe

      #REGION 0
     ┌IF m.talkstat = "ON"
     │   SET TALK ON
     └ENDIF
     ┌IF m.compstat = "ON"
     │   SET COMPATIBLE ON
     └ENDIF


      *    ┌──────────────────────────────────────────
  *       |                QUAL/MS-DOS Cleanup Code
  *       |
  *       |
  *       |_____________________________________________
  *

  #REGION 1
<-RETURN
  *----------------------------------------------------
  *
  * display popup notice
  *
  *----------------------------------------------------
ENDCASE


  *
  *
  *      _QSFOKOMJF          mbuttons VALID
  *
  *      Function Origin:
  *
  *      From Platform:      Windows
  *      From Screen:        QUAL,     Record Number:    2
  *      Variable:           mbuttons
  *      Called By:          VALID Clause
  *      Object Type:        Push Button
  *      Snippet Number:     1
  *
  *
  *
  FUNCTION _qsf0komjf      &&  mbuttons VALID
  #REGION 1
  DO CASE
  CASE mbuttons = 1   && edit
     IF !EMPTY(mq)
        DO edqual
     ENDIF
     mok = .T.
  CASE mbuttons = 2   && quit
     mok = .F.
     CLEAR READ
  ENDCASE
  SHOW GETS

  *
  *
  *      _QSFOKOMRX          mq VALID
  *
  *      Function Origin:
  *
  *      From Platform:      Windows
  *      From Screen:        QUAL,     Record Number:    3
  *      Variable:           mq
  *      Called By:          VALID Clause
  *      Object Type:        List
  *      Snippet Number:     2
  *
  *
  *
  FUNCTION _qsf0komrx     &&  mq VALID
```

```
  #REGION 1
  DO edqual

  *
  *
  *      _QSFOKON7E          mbuttons VALID
  *
  *      Function Origin:
  *
  *      From Platform:      MS-DOS
  *      From Screen:        QUAL,     Record Number:    7
  *      Variable:           mbuttons
  *      Called By:          VALID Clause
  *      Object Type:        Push Button
  *      Snippet Number:     3
  *
  *
  *
  FUNCTION _qsf0kon7e     &&  mbuttons VALID
  #REGION 1
  DO CASE
  CASE mbuttons = 1   && add
     DEACTIVATE WINDOW w_qe
     DO object.spr
     ACTIVATE WINDOW w_qe
     DO setquals
  CASE mbuttons = 2   && delete
     mdel = validobj()
     IF mdel
        = qualdel()
     ENDIF
  CASE mbuttons = 3   && ok
     mok = .T.
     CLEAR READ
  CASE mbuttons = 4   && cancel
     mok = .F.
     CLEAR READ
  ENDCASE
  SHOW GETS

  *
  *
  *      _QSFOKONBU          mq VALID
  *
  *      Function Origin:
  *
  *      From Platform:      MS-DOS
  *      From Screen:        QUAL,     Record Number:    8
  *      Variable:           mq
  *      Called By:          VALID Clause
  *      Object Type:        List
  *      Snippet Number:     4
  *
  *
  *
  FUNCTION _qsf0konbu     &&  mq VALID
  #REGION 1
  DO edqual


  *
  *
  *      QUAL/MS-DOS Supporting Procedures and Functions
  *
  *
  *
```

```

    #REGION 1

    *
    *
    *              QUAL Function POPUPSHOW
    *
    *
    *

    FUNCTION popupshow
    PARAMETERS errstr
   ┌IF NOT WEXIST("w_popnote")
   │   DEFINE WINDOW w_popnote ;
   │      FROM INT((SROW()-8)/2),INT((SCOL()-36)/2) ;
   │      TO INT((SROW()-8)/2)+7,INT((SCOL()-36)/2)+35 ;
   │      TITLE "One moment" ;
   │      FLOAT ;
   │      CLOSE ;
   │      SHADOW ;
   │      DOUBLE ;
   │      COLOR SCHEME 1
   └ENDIF
   ┌IF WVISIBLE("w_popnote")
   │  ACTIVATE WINDOW w_popnote SAME
   ├ELSE
   │  ACTIVATE WINDOW w_popnote NOSHOW
   └ENDIF
    @ 1,1 SAY errstr SIZE 3,31

   ┌IF NOT WVISIBLE("w_popnote")
   │  ACTIVATE WINDOW w_popnote
   └ENDIF
 <──RETURN ""


    *
    *
    *              QUAL Function POPUPHIDE
    *
    *
    *

    FUNCTION popuphide
    PARAMETERS W
    RELEASE WINDOW w_popnote
 <──RETURN W

    *------------------------------------
    *
    * This procedure validate the object to have access to delete or not.
    *
    *------------------------------------

    *
    *
    *              QUAL Function VALIDOBJ
    *
    *
    *

    FUNCTION validobj
    PRIVATE msel,mvalid
    mvalid = .F.
    msel = SELECT()
    SELECT quals
    SEEK mquals[mq,4]
   ┌IF FOUND()
   │ ┌IF EMPTY(rules) AND EMPTY(ruleso)
   │ │   mvalid = .T.
   │ ├ELSE
   │ │   msg = ALLTRIM(STRTRAN(rules,"|",","))
   │ │  ┌IF !EMPTY(ruleso)
   │ │  │   msg = msg + IIF(!EMPTY(msg), ",","") + ALLTRIM(STRTRAN(rules
=> o,"|",","))
   │ │  └ENDIF
   │ │  ┌IF LEN(msg) > 120
   │ │  │   msg = SUBSTR(msg,1,120)
   │ │  │   msg = SUBSTR(msg,1,RAT(",",msg)-1) + ", etc"
   │ │  └ENDIF
   │ │   mvalid = .F.
   │ │   =errmsg("This object was used by rules: " + msg + ".  Delete wa
=> s not allows!!.",2)
   │ └ENDIF
   └ENDIF
    SELECT (msel)
 <──RETURN mvalid


    *
    *
    *              QUAL Function QUALDEL
    *
    *
    *

    FUNCTION qualdel
    PRIVATE msel
    msel = SELECT()
    SELECT quals
    SEEK mquals[mq,4]
   ┌IF FOUND()
   │  = popupshow("deleting...")
   │  DELETE
   │  PACK
   │  = popuphide()
   └ENDIF
    DO setquals
    SELECT (msel)
 <──RETURN
  *:  EOF: QUAL.ac1
```

```
*
*
*     08/09/94                GOAL.SPR                09:39:04
*
*
*     Author's Name
*
*     Copyright (c) 1994 Company Name
*     Address
*     City,     Zip
*
*     Description:
*     This program was automatically generated by GENSCRN.
*
*

DO CASE
CASE _WINDOWS


   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
   ENDIF
   m.compstat = SET("COMPATIBLE")
   SET COMPATIBLE FOXPLUS

   m.rborder = SET("READBORDER")
   SET readborder ON

   *
   *
   *                Windows Window definitions
   *
   *
   *

   IF NOT WEXIST("w_goal") ;
         OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.PJX" ;
         OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.SCX" ;
         OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.MNX" ;
         OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.PRG" ;
         OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.FRX" ;
         OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.QPR"
      DEFINE WINDOW w_goal ;
         AT  0.000, 0.000  ;
         SIZE 22.500,48.375 ;
         TITLE "Goal Object Editor" ;
         FONT "Terminal", 8 ;
         FLOAT ;
         CLOSE ;
         SHADOW ;
         NOMINIMIZE ;
         COLOR RGB(,,,128,128,128)
      MOVE WINDOW w_goal CENTER
   ENDIF


   *
   *
   *            GOAL/Windows Screen Layout
   *
   *
   *

   #REGION 1
   IF WVISIBLE("w_goal")
      ACTIVATE WINDOW w_goal SAME
   ELSE
      ACTIVATE WINDOW w_goal NOSHOW
   ENDIF
   @ 20.167,17.000 GET mbuttons ;
      PICTURE "@*HN \<Edit;\<Quit" ;
      SIZE 1.917,6.000,0.750 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kopcm()
   @ 0.583,1.875 GET mg ;
      PICTURE "@&N" ;
      FROM mgoals ;
      SIZE 17.500,44.250 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kopg7()

   IF NOT WVISIBLE("w_goal")
      ACTIVATE WINDOW w_goal
   ENDIF

   READ CYCLE MODAL

   RELEASE WINDOW w_goal

   #REGION 0

   SET readborder &rborder

   IF m.talkstat = "ON"
      SET TALK ON
   ENDIF
   IF m.compstat = "ON"
      SET COMPATIBLE ON
   ENDIF


CASE _DOS


   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
```

```
  ENDIF
  m.compstat = SET("COMPATIBLE")
  SET COMPATIBLE FOXPLUS

  *
  *
  *             MS-DOS Window definitions
  *
  *

  *

 IF NOT WEXIST("w_goal") ;
        OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.PJX" ;
        OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.SCX" ;
        OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.MNX" ;
        OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.PRG" ;
        OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.FRX" ;
        OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.QPR"
    DEFINE WINDOW w_goal ;
        FROM INT((SROW()-19)/2),INT((SCOL()-64)/2) ;
        TO INT((SROW()-19)/2)+18,INT((SCOL()-64)/2)+63 ;
        TITLE "Goal Object Editor" ;
        FLOAT ;
        CLOSE ;
        SHADOW ;
        NOMINIMIZE ;
        COLOR SCHEME 1
 ENDIF


  *
  *
  *             GOAL/MS-DOS Screen Layout
  *
  *

  *

  #REGION 1
 IF WVISIBLE("w_goal")
    ACTIVATE WINDOW w_goal SAME
 ELSE
    ACTIVATE WINDOW w_goal NOSHOW
 ENDIF
  @ 16,17 GET mbuttons ;
    PICTURE "@*HN \<Add;\<OK;\<Cancel" ;
    SIZE 1,8,1 ;
    DEFAULT 1 ;
    VALID _qsf0koptk()
  @ 1,1 GET mg ;
    PICTURE "@&N" ;
    FROM mgoals ;
    SIZE 14,60 ;
    DEFAULT 1 ;
    VALID _qsf0kopxh() ;
    COLOR SCHEME 2

  IF NOT WVISIBLE("w_goal")
     ACTIVATE WINDOW w_goal
  ENDIF

  READ CYCLE MODAL

  RELEASE WINDOW w_goal

  #REGION 0
  IF m.talkstat = "ON"
     SET TALK ON
  ENDIF
  IF m.compstat = "ON"
     SET COMPATIBLE ON
  ENDIF


ENDCASE


  *
  *      _QSFOKOPCM          mbuttons VALID
  *
  *      Function Origin:
  *
  *      From Platform:      Windows
  *      From Screen:        GOAL,     Record Number:    2
  *      Variable:           mbuttons
  *      Called By:          VALID Clause
  *      Object Type:        Push Button
  *      Snippet Number:     1
  *
  *
  *
  FUNCTION _qsf0kopcm     &&  mbuttons VALID
  #REGION 1
 DO CASE
 CASE mbuttons = 1   && edit
   IF !EMPTY(mg)
      DO edgoal
   ENDIF
 CASE mbuttons = 2   && quit
   mok = .F.
   CLEAR READ
 ENDCASE
  SHOW GETS


  *
  *
  *      _QSFOKOPG7          mg VALID
  *
  *      Function Origin:
  *
  *      From Platform:      Windows
  *      From Screen:        GOAL,     Record Number:    3
  *      Variable:           mg
  *      Called By:          VALID Clause
  *      Object Type:        List
  *      Snippet Number:     2
  *
  *
  *
  FUNCTION _qsf0kopg7     &&  mg VALID
  #REGION 1
```

```
DO edgoal

*
*
*        _QSFOKOPTK          mbuttons VALID
*
*        Function Origin:
*
*        From Platform:      MS-DOS
*        From Screen:        GOAL,     Record Number:    6
*        Variable:           mbuttons
*        Called By:          VALID Clause
*        Object Type:        Push Button
*        Snippet Number:     3
*
*
*
FUNCTION _qsf0koptk     &&  mbuttons VALID
#REGION 1
DO CASE
CASE mbuttons = 1
   DEACTIVATE WINDOW w_goal
   DO object.spr WITH "G"
   ACTIVATE WINDOW w_goal
   DO setgoals
CASE mbuttons = 2   && ok
   mok = .T.
   CLEAR READ
CASE mbuttons = 3   && cancel
   mok = .F.
   CLEAR READ
ENDCASE
SHOW GETS


*
*
*        _QSFOKOPXH          mg VALID
*
*        Function Origin:
*
*        From Platform:      MS-DOS
*        From Screen:        GOAL,     Record Number:    7
*        Variable:           mg
*        Called By:          VALID Clause
*        Object Type:        List
*        Snippet Number:     4
*
*
*
FUNCTION _qsf0kopxh     &&  mg VALID
#REGION 1
DO edgoal
*:  EOF: GOAL.ac1
```

1C/14PRINT

```
*
*
*       08/09/94             DISPLAY.SPR              09:39:07
*
*
*       Author's Name
*
*       Copyright (c) 1994 Company Name
*       Address
*       City,      Zip
*
*       Description:
*       This program was automatically generated by GENSCRN.
*
*

DO CASE
CASE _WINDOWS


   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
   ENDIF
   m.compstat = SET("COMPATIBLE")
   SET COMPATIBLE FOXPLUS

   m.rborder = SET("READBORDER")
   SET readborder ON

   *
   *
   *              Windows Window definitions
   *
   *
   *


   IF NOT WEXIST("w_displ") ;
         OR UPPER(WTITLE("W_DISPL")) == "W_DISPL.PJX" ;
         OR UPPER(WTITLE("W_DISPL")) == "W_DISPL.SCX" ;
         OR UPPER(WTITLE("W_DISPL")) == "W_DISPL.MNX" ;
         OR UPPER(WTITLE("W_DISPL")) == "W_DISPL.PRG" ;
         OR UPPER(WTITLE("W_DISPL")) == "W_DISPL.FRX" ;
         OR UPPER(WTITLE("W_DISPL")) == "W_DISPL.QPR"
      DEFINE WINDOW w_displ ;
         AT  0.000, 0.000  ;
         SIZE 21.250,48.500 ;
         TITLE "Display Object Editor" ;
         FONT "Terminal", 8 ;
         FLOAT ;
         CLOSE ;
         SHADOW ;
         NOMINIMIZE ;
         COLOR RGB(,,,128,128,128)
      MOVE WINDOW w_displ CENTER
   ENDIF


   *
   *
   *              DISPLAY/Windows Screen Layout
   *
   *
   *

   #REGION 1
   IF WVISIBLE("w_displ")
      ACTIVATE WINDOW w_displ SAME
   ELSE
      ACTIVATE WINDOW w_displ NOSHOW
   ENDIF
   @ 18.833,16.000 GET mbuttons ;
      PICTURE "@*HT \<Edit;\<Quit" ;
      SIZE 1.917,6.000,0.750 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0korj1()
   @ 0.667,1.750 GET md ;
      PICTURE "@&N" ;
      FROM mdispl ;
      SIZE 16.333,44.250 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kormj()

   IF NOT WVISIBLE("w_displ")
      ACTIVATE WINDOW w_displ
   ENDIF

   READ CYCLE MODAL

   RELEASE WINDOW w_displ

   #REGION 0

   SET readborder &rborder

   IF m.talkstat = "ON"
      SET TALK ON
   ENDIF
   IF m.compstat = "ON"
      SET COMPATIBLE ON
   ENDIF


CASE _DOS


   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
```

```
ENDIF
m.compstat = SET("COMPATIBLE")
SET COMPATIBLE FOXPLUS

*
*
*        MS-DOS Window definitions
*
*
*

IF NOT WEXIST("w_displ") ;
      OR UPPER(WTITLE("W_DISPL")) == "W_DISPL.PJX" ;
      OR UPPER(WTITLE("W_DISPL")) == "W_DISPL.SCX" ;
      OR UPPER(WTITLE("W_DISPL")) == "W_DISPL.MNX" ;
      OR UPPER(WTITLE("W_DISPL")) == "W_DISPL.PRG" ;
      OR UPPER(WTITLE("W_DISPL")) == "W_DISPL.FRX" ;
      OR UPPER(WTITLE("W_DISPL")) == "W_DISPL.QPR"
   DEFINE WINDOW w_displ ;
      FROM INT((SROW()-18)/2),INT((SCOL()-64)/2) ;
      TO INT((SROW()-18)/2)+17,INT((SCOL()-64)/2)+63 ;
      TITLE "Display Object Editor" ;
      FLOAT ;
      CLOSE ;
      SHADOW ;
      NOMINIMIZE ;
      COLOR SCHEME 1
ENDIF


*
*
*        DISPLAY/MS-DOS Screen Layout
*
*
*

#REGION 1
IF WVISIBLE("w_displ")
   ACTIVATE WINDOW w_displ SAME
ELSE
   ACTIVATE WINDOW w_displ NOSHOW
ENDIF
@ 15,29 GET mbuttons ;
   PICTURE "@*HT OK;Cancel" ;
   SIZE 1,8,1 ;
   DEFAULT 1 ;
   VALID _qsf0korzx()
@ 15,11 GET mbbutton ;
   PICTURE "@*HN Add;Delete" ;
   SIZE 1,8,1 ;
   DEFAULT 1 ;
   VALID _qsf0kos30()
@ 0,1 GET md ;
   PICTURE "@&N" ;
   FROM mdispl ;
      SIZE 14,60 ;
      DEFAULT 1 ;
      VALID _qsf0kos6a() ;
      COLOR SCHEME 2

   IF NOT WVISIBLE("w_displ")
      ACTIVATE WINDOW w_displ
   ENDIF

   READ CYCLE MODAL

   RELEASE WINDOW w_displ

   #REGION 0
   IF m.talkstat = "ON"
      SET TALK ON
   ENDIF
   IF m.compstat = "ON"
      SET COMPATIBLE ON
   ENDIF

ENDCASE


*
*
*      _QSFOKORJ1            mbuttons VALID
*
*      Function Origin:
*
*      From Platform:        Windows
*      From Screen:          DISPLAY,     Record Number:   2
*      Variable:             mbuttons
*      Called By:            VALID Clause
*      Object Type:          Push Button
*      Snippet Number:       1
*
*
*
FUNCTION _qsf0korj1     &&  mbuttons VALID
#REGION 1
DO CASE
CASE mbuttons = 1   && Edit
   IF !EMPTY(md)
      DO eddisplay
   ENDIF
   mok = .T.
CASE mbuttons = 2   && Quit
   mok = .F.
   CLEAR READ
ENDCASE

*
*
*      _QSFOKORMJ            md VALID
*
*      Function Origin:
*
*      From Platform:        Windows
*      From Screen:          DISPLAY,     Record Number:   3
*      Variable:             md
*      Called By:            VALID Clause
*      Object Type:          List
*      Snippet Number:       2
*
```

```
*
*
FUNCTION _qsf0kormj      &&  md VALID
#REGION 1
DO eddisplay

*
*         _QSFOKORZX          mbuttons VALID
*
*         Function Origin:
*
*         From Platform:      MS-DOS
*         From Screen:        DISPLAY,     Record Number:   6
*         Variable:           mbuttons
*         Called By:          VALID Clause
*         Object Type:        Push Button
*         Snippet Number:     3
*
*
*
FUNCTION _qsf0korzx      &&  mbuttons VALID
#REGION 1
DO CASE
CASE mbuttons = 1   && ok
   mok = .T.
CASE mbuttons = 2   && cancel
   mok = .F.
ENDCASE

*
*
*         _QSFOKOS30          mbbutton VALID
*
*         Function Origin:
*
*         From Platform:      MS-DOS
*         From Screen:        DISPLAY,     Record Number:   7
*         Variable:           mbbutton
*         Called By:          VALID Clause
*         Object Type:        Push Button
*         Snippet Number:     4
*
*
*
FUNCTION _qsf0kos30      &&  mbbutton VALID
#REGION 1
DO CASE
CASE mbbutton = 1
   APPEND BLANK
   REPLACE area WITH area.area
CASE mbbutton = 2
ENDCASE
SHOW GETS


*
*
*         _QSFOKOS6A          md VALID
*
*         Function Origin:
*
*         From Platform:      MS-DOS
*         From Screen:        DISPLAY,     Record Number:   8
*         Variable:           md
*         Called By:          VALID Clause
*         Object Type:        List
*         Snippet Number:     5
*
*
*
FUNCTION _qsf0kos6a      &&  md VALID
#REGION 1
DO eddisplay
*:  EOF: DISPLAY.ac1
```

I-3

```
*
*
*       08/09/94            DISEASE.SPR            09:39:10
*
*
*       Author's Name
*
*       Copyright (c) 1994 Company Name
*       Address
*       City,     Zip
*
*       Description:
*       This program was automatically generated by GENSCRN.
*
*

PARAMETERS m.adding
DO CASE
CASE _WINDOWS


   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
   ENDIF
   m.compstat = SET("COMPATIBLE")
   SET COMPATIBLE FOXPLUS

   m.rborder = SET("READBORDER")
   SET readborder ON

   *
   *
   *              Windows Window definitions
   *
   *
   *

   IF NOT WEXIST("w_disease") ;
         OR UPPER(WTITLE("W_DISEASE")) == "W_DISEASE.PJX" ;
         OR UPPER(WTITLE("W_DISEASE")) == "W_DISEASE.SCX" ;
         OR UPPER(WTITLE("W_DISEASE")) == "W_DISEASE.MNX" ;
         OR UPPER(WTITLE("W_DISEASE")) == "W_DISEASE.PRG" ;
         OR UPPER(WTITLE("W_DISEASE")) == "W_DISEASE.FRX" ;
         OR UPPER(WTITLE("W_DISEASE")) == "W_DISEASE.QPR"
      DEFINE WINDOW w_disease ;
         AT  0.000, 0.000  ;
         SIZE 28.333,66.500 ;
         TITLE "Disease Object Editor" ;
         FONT "Terminal", 8 ;
         FLOAT ;
         CLOSE ;
         SHADOW ;
         NOMINIMIZE ;
         COLOR RGB(,,,0,255,255)
      MOVE WINDOW w_disease CENTER
   ENDIF


   *
   *
   *              DISEASE/Windows Setup Code - SECTION 2
   *
   *
   *

   #REGION 1
   PRIVATE msel
   msel = SELECT()
   SELECT disease
   SCATTER MEMVAR MEMO

   *
   *
   *              DISEASE/Windows Screen Layout
   *
   *
   *

   #REGION 1
   IF WVISIBLE("w_disease")
      ACTIVATE WINDOW w_disease SAME
   ELSE
      ACTIVATE WINDOW w_disease NOSHOW
   ENDIF
   @ 0.333,15.250 SAY "Name"  ;
      FONT "Terminal", 8
   @ 0.417,2.875 SAY "Id"  ;
      FONT "Terminal", 8
   @ 13.417,2.625 SAY "Treatment"  ;
      FONT "Terminal", 8 ;
      STYLE "T"
   @ 1.833,2.750 SAY "Description:"  ;
      FONT "Terminal", 8 ;
      STYLE "T"
   @ 0.500,5.875 GET m.id ;
      SIZE 1.000,5.000 ;
      DEFAULT " " ;
      FONT "Terminal", 8 ;
      DISABLE
   @ 0.417,20.625 GET m.name ;
      SIZE 1.000,41.125 ;
      DEFAULT " " ;
      FONT "Terminal", 8 ;
      PICTURE "@!" ;
      DISABLE
   @ 3.500,2.875 EDIT m.descript ;
      SIZE 9.250,61.000,0.000 ;
      DEFAULT " " ;
      FONT "Terminal", 8 ;
      SCROLL
```

```
@ 14.833,2.625 EDIT m.treatment ;
   SIZE 10.417,61.000,0.000 ;
   DEFAULT " " ;
   FONT "Terminal", 8 ;
   SCROLL
@ 25.917,25.750 GET mbuttons ;
   PICTURE "@*HT \<OK;\<Cancel" ;
   SIZE 1.917,7.500,0.750 ;
   DEFAULT 1 ;
   FONT "Terminal", 8 ;
   VALID _qsf0kou0d()

IF NOT WVISIBLE("w_disease")
   ACTIVATE WINDOW w_disease
ENDIF

READ CYCLE MODAL ;
   WHEN _qsf0kou3p()

RELEASE WINDOW w_disease

#REGION 0

SET readborder &rborder

IF m.talkstat = "ON"
   SET TALK ON
ENDIF
IF m.compstat = "ON"
   SET COMPATIBLE ON
ENDIF

*
*
*              DISEASE/Windows Cleanup Code
*
*
*

#REGION 1
SELECT (msel)

CASE _DOS

*
*
*              DISEASE/MS-DOS Setup Code - SECTION 1
*
*
*

#REGION 1
IF PARAMETER() = 0
   m.adding = 0
ENDIF

#REGION 0
REGIONAL m.currarea, m.talkstat, m.compstat

IF SET("TALK") = "ON"
   SET TALK OFF
   m.talkstat = "ON"
ELSE
   m.talkstat = "OFF"
ENDIF
m.compstat = SET("COMPATIBLE")
SET COMPATIBLE FOXPLUS

*
*
*              MS-DOS Window definitions
*
*
*

IF NOT WEXIST("w_disease") ;
      OR UPPER(WTITLE("W_DISEASE")) == "W_DISEASE.PJX" ;
      OR UPPER(WTITLE("W_DISEASE")) == "W_DISEASE.SCX" ;
      OR UPPER(WTITLE("W_DISEASE")) == "W_DISEASE.MNX" ;
      OR UPPER(WTITLE("W_DISEASE")) == "W_DISEASE.PRG" ;
      OR UPPER(WTITLE("W_DISEASE")) == "W_DISEASE.FRX" ;
      OR UPPER(WTITLE("W_DISEASE")) == "W_DISEASE.QPR"
   DEFINE WINDOW w_disease ;
      FROM INT((SROW()-20)/2),INT((SCOL()-78)/2) ;
      TO INT((SROW()-20)/2)+19,INT((SCOL()-78)/2)+77 ;
      TITLE "Disease Object Editor" ;
      FLOAT ;
      CLOSE ;
      SHADOW ;
      NOMINIMIZE ;
      COLOR SCHEME 1
ENDIF


*
*
*              DISEASE/MS-DOS Screen Layout
*
*
*

#REGION 1
IF WVISIBLE("w_disease")
   ACTIVATE WINDOW w_disease SAME
ELSE
   ACTIVATE WINDOW w_disease NOSHOW
ENDIF
@ 0,19 GET disease.name ;
   SIZE 1,35 ;
   DEFAULT " " ;
```

```
         PICTURE "@!" ;
         WHEN m.adding = disease.id ;
         DISABLE
      @ 0,13 SAY "Name" ;
         SIZE 1,4, 0
      @ 1,0 EDIT disease.descript ;
         SIZE 7,76,0 ;
         DEFAULT " " ;
         SCROLL
      @ 9,0 EDIT disease.treatment ;
         SIZE 7,76,0 ;
         DEFAULT " " ;
         SCROLL
      @ 17,29 GET mbuttons ;
         PICTURE "@*HN \<OK;\<Cancel" ;
         SIZE 1,8,1 ;
         DEFAULT 1 ;
         VALID _qsf0koumi()
      @ 0,0 SAY "Id" ;
         SIZE 1,2, 0
      @ 0,4 GET disease.id ;
         SIZE 1,5 ;
         DEFAULT " " ;
         DISABLE

      IF NOT WVISIBLE("w_disease")
         ACTIVATE WINDOW w_disease
      ENDIF

      READ CYCLE MODAL ;
         WHEN _qsf0kouqk()

      RELEASE WINDOW w_disease

      #REGION 0
      IF m.talkstat = "ON"
         SET TALK ON
      ENDIF
      IF m.compstat = "ON"
         SET COMPATIBLE ON
      ENDIF

   ENDCASE


      *
      *
      *       _QSFOKOUOD          mbuttons VALID
      *
      *       Function Origin:
      *
      *       From Platform:      Windows
      *       From Screen:        DISEASE,      Record Number:   10
      *       Variable:           mbuttons
      *       Called By:          VALID Clause
      *       Object Type:        Push Button
      *       Snippet Number:     1
      *
      *
      *
      FUNCTION _qsf0kou0d     &&  mbuttons VALID
      #REGION 1
      DO CASE
      CASE mbuttons = 1    && ok
         SELECT disease
         GATHER MEMVAR MEMO
         mok = .T.
      CASE mbuttons = 2    && cancel
         mok = .F.
      ENDCASE
      SHOW GETS

      *
      *
      *       _QSFOKOU3P          Read Level When
      *
      *       Function Origin:
      *
      *
      *       From Platform:      Windows
      *       From Screen:        DISEASE
      *       Called By:          READ Statement
      *       Snippet Number:     2
      *
      *
      *
      FUNCTION _qsf0kou3p     && Read Level When
      *
      * When Code from screen: DISEASE
      *
      #REGION 1
      SET TOPIC TO "GOAL"
      SHOW GET m.name ENABLE
      _CUROBJ = OBJNUM(m.name)
      SHOW GETS


      *
      *
      *       _QSFOKOUMI          mbuttons VALID
      *
      *       Function Origin:
      *
      *       From Platform:      MS-DOS
      *       From Screen:        DISEASE,      Record Number:   17
      *       Variable:           mbuttons
      *       Called By:          VALID Clause
      *       Object Type:        Push Button
      *       Snippet Number:     3
      *
      *
      *
      FUNCTION _qsf0koumi     &&  mbuttons VALID
      #REGION 1
      DO CASE
      CASE mbuttons = 1    && ok
         mok = .T.
         CLEAR READ
      CASE mbuttons = 2    && cancel
         mok = .F.
         CLEAR READ
      ENDCASE
      SHOW GETS

      *
      *
      *       _QSFOKOUQK          Read Level When
      *
      *       Function Origin:
      *
      *
```

```
*        From Platform:      MS-DOS
*        From Screen:        DISEASE
*        Called By:          READ Statement
*        Snippet Number:     4
*
*
*
FUNCTION _qsf0kouqk     && Read Level When
*
* When Code from screen: DISEASE
*
#REGION 1
SET TOPIC TO "GOAL"
SHOW GET disease.name ENABLE
_CUROBJ = OBJNUM(disease.name)
SHOW GETS



*
*
*         DISEASE/MS-DOS Supporting Procedures and Functions
*
*
*

#REGION 1
*:  EOF: DISEASE.ac1
```

```
*:***********************************************************************
=> *******
*:
*: Procedure file: C:\CAMD2\KBEDIT\WORKW\KBLDR.PRG
*:         System: Knowledge Base Editor
*:         Author: Hoa L. Ly
*:      Copyright (c) June 1, 1994, Naval Health Research Center, Code 2
=> 2
*:  Last modified: 08/09/94 at  8:37:34
*:
*:         Set by: _QSFOKQC6T()       (function in KBLOAD.SPR)
*:               : _QSFOKQCZG()       (function in KBLOAD.SPR)
*:
*:           Uses: FACT.DBF
*:               : PREMISE.DBF
*:               : ACTION.DBF
*:               : RULE.DBF
*:               : KBHELP.DBF              Alias: HELP
*:
*:        Indexes: FACT.IDX
*:               : PREMISE.IDX
*:               : ACTION.IDX
*:               : RULEAREA.IDX
*:               : SALIENCE.IDX
*:               : RULE.IDX
*:
*:      Documented 15:00:57                                  FoxDoc versio
=> n 3.00a
*:***********************************************************************
=> *******
    * kbldr.prg
    *-------------------------------------------------------
    *
    *                    Knowledge Base Loader
    *                             for
    *               Medical Practice Support System
    *
    *                       $Revision: 1.1 $
    *                   $Date: 92/05/04 11:10:24 $
    *                     $Author: RoyWDobbins $
    *
    *-------------------------------------------------------
    PARAMETERS m.new                        && source directory

    * Load a new knowledge base

    * Objects from the new knowledge base are appended to the
    * existing knowledge base files, or replace existing objects.
    * Existing rulesets and their associated premise, action clauses
    * are deleted first, before the new ruleset is loaded
    * Other objects may only be edited, not replaced.

    PRIVATE m.exists                        && true if item exists
    PRIVATE m.unique                        && enumerated item
    PRIVATE m.uord                          && next ordinal value
    PRIVATE mexact                          && set exact on/off
    PRIVATE m.newrecno                                          && stor
=> e current area record number

    * check all the database exist at m.new directory.
    * If any database was not define will be stop process.
  ┌─IF !FILE(m.new + "dict.dbf")
<──────RETURN
  └─ENDIF
  ┌─IF !FILE(m.new + "val.dbf")
<──────RETURN
  └─ENDIF
  ┌─IF !FILE(m.new + "enum.dbf")
<──────RETURN
  └─ENDIF
  ┌─IF !FILE(m.new + "disease.dbf")
<──────RETURN
  └─ENDIF
  ┌─IF !FILE(m.new + "help.dbf")
<──────RETURN
  └─ENDIF
  ┌─IF !FILE(m.new + "area.dbf")
<──────RETURN
  └─ENDIF
  ┌─IF !FILE(m.new + "fact.dbf")
<──────RETURN
  └─ENDIF
  ┌─IF !FILE(m.new + "premise.dbf")
<──────RETURN
  └─ENDIF
  ┌─IF !FILE(m.new + "action.dbf")
<──────RETURN
  └─ENDIF
  ┌─IF !FILE(m.new + "rule.dbf")
<──────RETURN
  └─ENDIF
  ┌─IF !FILE(m.new + "goals.dbf")
<──────RETURN
  └─ENDIF
  ┌─IF !FILE(m.new + "display.dbf")
<──────RETURN
  └─ENDIF

    * Open the database
    SELECT 0
    USE fact INDEX fact
    SELECT 0
    USE premise INDEX premise
    SET RELATION TO fact INTO fact
    SELECT 0
    USE action INDEX action
    SELECT 0
    USE rule INDEX rulearea,salience,rule
    SET RELATION TO premise INTO premise, action INTO action
    SELECT area
    SET RELATION TO area INTO rule

    SELECT area
    m.newrecno = RECNO()

    *-------------------------------------------------------
    *
    * update dict/enum/val databases
    *
    *-------------------------------------------------------
    SELECT dict
    GOTO BOTTOM
    m.qualid = id + 1                         && next available id
    SET ORDER TO 2                            && use name key
    SELECT 0
    USE (m.new + "dict") ALIAS newdict        &&

    i = MAX( RECCOUNT(), 1 )
    PRIVATE mqid, mquid
    DIMENSION mqid[i], mquid[i]

  ┏DO WHILE !EOF()
```

```
    SCATTER MEMVAR                          && load new dict element
    m.newid = m.id                          && new item id
    mqid[ RECNO() ] = m.id
    SELECT dict
    m.name = UPPER( TRIM( m.name ) )        && new item name
    IF LEN( m.name ) > LEN( name )
       m.name = SUBSTR( m.name, LEN( name ) )
    ENDIF
    SEEK m.name
    IF FOUND()
       * item already exists
       * some fields cannot be edited - use existing values
       m.datatype = datatype
       m.exists = .T.
       m.id = id
    ELSE
       m.exists = .F.
       m.id = m.id + m.qualid               && use next available id
       APPEND BLANK                         && add a new dict element
    ENDIF
    GATHER MEMVAR                           && update dict record
    m.uid = m.id                            && updated item id
    IF m.datatype = "N"
       * update numeric type
       SELECT 0
       USE (m.new + "val") ALIAS newval
       LOCATE FOR id = m.newid
       SCATTER MEMVAR                       && load new val
       SELECT VAL
       SEEK m.uid
       IF !FOUND()
          APPEND BLANK                      && add a new val
       ENDIF
       m.id = m.uid
       GATHER MEMVAR                        && update the val record
       SELECT newval
       USE
    ELSE
       * update enumerated types
       * You can only append additional enumerated values
       SELECT enum
       SEEK m.uid
       m.exists = FOUND()                   && see if record(s) already
=> there
       m.uord = 0
       IF m.exists
          * next available ordinal value
          COUNT WHILE id = m.uid .AND. !EOF() TO m.uord
       ENDIF
       SELECT 0
       USE (m.new + "enum") ALIAS newenum
       LOCATE FOR id = m.newid
       DO WHILE id = m.newid .AND. !EOF()
          SCATTER MEMVAR                    && load new enum
          m.unique = .T.
          IF m.exists
             * existing element, ensure enumeration is new
             SELECT enum
             SEEK m.uid
             mexact = SET("EXACT")
             SET EXACT ON                   && check exact matches
             DO WHILE id = m.uid .AND. !EOF()
                IF enumerate = m.enumerate
                   m.unique = .F.           && already got this
             v-----EXIT
                ENDIF
                SKIP
             ENDDO
             IF mexact = "OFF"
                SET EXACT OFF
             ENDIF
             SELECT newenum
          ENDIF
          IF m.unique
             SELECT enum
             APPEND BLANK                   && add a new enum
             m.id = m.uid
             m.uord = m.uord + 1            && assign next ordinal value
             m.ord = m.uord
             GATHER MEMVAR                  && update enum record
             SELECT newenum
          ENDIF
          SKIP
       ENDDO
       USE
    ENDIF
    SELECT newdict
    mquid[recno()] = m.uid                  && update id
    SKIP
 ENDDO
 USE
 SELECT dict
 SET ORDER TO 1                             && restore primary index

 *-----------------------------------------------------------------
 *
 * update disease file
 *
 *-----------------------------------------------------------------
 SELECT disease
 GOTO BOTTOM
 m.newid = id                               && next available id
 m.newid = MAX(m.newid, 10000)              && starting at 10000
 m.newid = m.newid + 1
 SET ORDER TO 2                             && name order

 SELECT 0
 USE (m.new + "disease") ALIAS newdis

 i = MAX( RECCOUNT(), 1 )
 PRIVATE mgid, mguid
 DIMENSION mgid[ i ],  mguid[ i ]

 DO WHILE !EOF()
    SCATTER MEMVAR MEMO                     && load new disease
    mgid[ RECNO() ] = m.id
    SELECT disease
    m.name = UPPER( TRIM( m.name ) )        && new item name
    IF LEN( m.name ) > LEN( name )
       m.name = SUBSTR( m.name, LEN( name ) )
    ENDIF
    SEEK m.name
    IF FOUND()
       m.exists = .T.
       m.id = id
       m.name = name
    ELSE
       m.exists = .F.
       m.id = m.id + m.newid
       APPEND BLANK                         && add a new disease
    ENDIF
    GATHER MEMVAR MEMO                      && update the disease record
```

```
   │  SELECT newdis
   │  mguid[recno()] = m.id              && update id
   │  SKIP
   └ENDDO
   USE
   SELECT disease
   SET ORDER TO 1


   *-----------------------------------------------------
   *
   * update help file
   *
   *-----------------------------------------------------
   SELECT 0
   USE kbhelp ALIAS HELP
   SELECT 0
   USE (m.new + "help") ALIAS newhelp

  ┌DO WHILE !EOF()
  │  SCATTER MEMVAR                      && load new help
  │  m.details = details
  │  m.topic = UPPER(m.topic)
  │  SELECT HELP
  │  LOCATE FOR m.topic = UPPER(TOPIC)
  │ ┌IF !FOUND()
  │ │   APPEND BLANK                     && add a new help
  │ └ENDIF
  │  GATHER MEMVAR                       && update the help record
  │  REPLACE details WITH m.details
  │  SELECT newhelp
  │  SKIP
  └ENDDO
   USE
   SELECT HELP
   USE


   *-----------------------------------------------------
   *
   * Get the new knowledge base descriptor
   *
   *-----------------------------------------------------
   SELECT 0
   USE (m.new + "area") ALIAS newarea
   SCATTER MEMVAR                        && load the record
   USE

   * Lookup knowledge base name (case insensitive)
   SELECT area
   LOCATE FOR LOWER(TRIM(name)) = LOWER( TRIM( m.name ) )   && lookup by
=>  name
  ┌IF FOUND()
  │  m.area = area
  ├ELSE
  │  * new knowledge base
  │  GOTO BOTTOM
  │  m.area = area + 1                   && get new id
  │  APPEND BLANK
  └ENDIF
   m.newrecno = RECNO()
   GATHER MEMVAR                         && import new record

   *-----------------------------------------------------
   *
   * delete existing rules in knowledge base
   *
   *-----------------------------------------------------
   SELECT rule
   SEEK m.area
  ┌DO WHILE area = m.area .AND. !EOF()
  │  SELECT premise
  │  SEEK rule.premise
  │ ┌DO WHILE clause = rule.premise
  │ │ ┌IF EMPTY( premise.op )
  │ │ │   SELECT fact
  │ │ │   DELETE FOR clause = premise.fact   && delete all premise cla
=> uses
  │ │ └ENDIF
  │ │  SELECT premise
  │ │  DELETE
  │ │  SKIP
  │ └ENDDO
  │  SELECT action
  │  DELETE FOR clause = rule.action      && delete all action clauses
  │  DELETE FOR clause = rule.else        && delete all else clauses
  │  SELECT rule
  │  DELETE                              && delete the rule
  │  SKIP
  └ENDDO

   *-----------------------------------------------------
   *
   * prepare fact database
   *
   *-----------------------------------------------------
   SELECT fact
   PACK                                  && purge deleted facts
   GOTO BOTTOM
   m.fact = clause                       && get last clause

   * update fact clauses
   SELECT 0
   USE (m.new + "fact")
   COPY TO "tempxxxx"                    && renumber clauses
   USE ("tempxxxx")
   REPLACE ALL clause WITH m.fact + clause
   REPLACE ALL id WITH ;
      IIF(OBJECT = "D", mguid[ ASCAN(mgid, id) ], ;
      mquid[ ASCAN(mqid, id) ] )
   * update any indirect references
   GOTO TOP

  ┌DO WHILE !EOF()
  │  m.val = VAL
  │ ┌FOR i = 1 TO 2
  │ │   K = AT( "@", m.val, i )
  │ │  ┌IF K = 0
  │ v──────EXIT
  │ │  └ENDIF
  │ │   m.id     =  VAL ( SUBSTR( m.val, K + 1 ) )
  │ │   m.newid  =  mquid[ ASCAN( mqid, m.id ) ]
  │ │   m.val = STRTRAN(  m.val, LTRIM( STR( m.id ) ), LTRIM( STR( m.ne
=> wid ) )  )
  │ └NEXT
  │  REPLACE VAL WITH m.val
  │  SKIP
  └ENDDO

   USE
   SELECT fact
   APPEND FROM ("tempxxxx")              && load new facts
```

```
DELETE FILE "tempxxxx.dbf"                 && remove temporary files

*-----------------------------------------------------
*
* prepare premise database
*
*-----------------------------------------------------
SELECT premise
PACK                                       && purge deleted premises
GOTO BOTTOM
m.premise = clause                         && get last clause

* update premises clauses
SELECT 0
USE (m.new + "premise")
COPY TO "tempxxxx"                         && renumber clauses
USE ("tempxxxx")
REPLACE ALL clause WITH m.premise + clause
REPLACE ALL fact WITH m.fact + fact  FOR EMPTY(op)
USE
SELECT premise
APPEND FROM ("tempxxxx")                   && load new premises
DELETE FILE "tempxxxx.dbf"                 && remove temporary files

*-----------------------------------------------------
*
* prepare action database
*
*-----------------------------------------------------
SELECT action
PACK                                       && purge delete actions
GOTO BOTTOM
m.action = clause                          && get last clause

* update action clauses
SELECT 0
USE (m.new + "action")
COPY TO "tempxxxx"                         && renumber clauses
USE ("tempxxxx")
REPLACE ALL ;
   clause WITH m.action + clause
REPLACE ALL id WITH ;
   IIF(OBJECT = "D", mguid[ ASCAN(mgid, id) ], ;
   mquid[ ASCAN(mqid, id) ] )
* update any indirect references
GOTO TOP

┌DO WHILE !EOF()
│  m.val = VAL
│ ┌FOR i = 1 TO 2
│ │   K = AT( "@", m.val, i )
│ │  ┌IF K = 0
│ v──────EXIT
│ │  └ENDIF
│ │   m.id     =  VAL ( SUBSTR( m.val, K + 1 ) )
│ │   m.newid  =  mquid[ ASCAN( mqid, m.id ) ]
│ │   m.val = STRTRAN(  m.val, LTRIM( STR( m.id ) ), LTRIM( STR( m.ne
=> wid ) ) )
│ └NEXT
│  REPLACE VAL WITH m.val
│  SKIP
└ENDDO

 USE
 SELECT action
 APPEND FROM ("tempxxxx")                  && load new actions
DELETE FILE "tempxxxx.dbf"                 && remove temporary files

*-----------------------------------------------------
*
* update rulebase
*
*-----------------------------------------------------
* Count the rules in new knowledge base
SELECT 0
USE (m.new + "rule")
REPLACE area.rules WITH RECCOUNT()         && update record with rule c
=> ount
COPY TO "tempxxxx"
USE ("tempxxxx")
REPLACE ALL ;
   premise WITH m.premise + premise, ;
   action WITH m.action + action, ;
   ELSE WITH IIF( ELSE > 0, m.action + ELSE, 0 ), ;
   area WITH m.area
USE

SELECT rule
PACK                                       && purge deleted rules
APPEND FROM ("tempxxxx")                   && load the new rules
DELETE FILE "tempxxxx.dbf"                 && remove temporary files
┌IF FILE("tempxxxx.fpt")
│  DELETE FILE "tempxxxx.fpt"
└ENDIF

*-----------------------------------------------------
*
* prepare goals
*
*-----------------------------------------------------
SELECT 0
USE (m.new + "goals")
COPY TO "tempxxxx"
USE ("tempxxxx")
REPLACE ALL id WITH ;
   IIF( OBJECT = "D",  mguid[ ASCAN( mgid, id ) ], ;
   mquid[ ASCAN( mqid, id ) ] )
REPLACE ALL area WITH m.area
USE
SELECT goals
DELETE FOR area = m.area
PACK
APPEND FROM ("tempxxxx")
DELETE FILE "tempxxxx.dbf"

*-----------------------------------------------------
*
* prepare display list
*
*-----------------------------------------------------
SELECT 0
USE (m.new + "display")
COPY TO "tempxxxx"
USE ("tempxxxx")
REPLACE ALL id WITH ;
   IIF( OBJECT = "D",  mguid[ ASCAN( mgid, id ) ], ;
   mquid[ ASCAN( mqid, id ) ] )
REPLACE ALL area WITH m.area
USE
SELECT DISPLAY
DELETE FOR area = m.area
PACK
```

1C/14PRINT

```
APPEND FROM ("tempxxxx")
DELETE FILE "tempxxxx.dbf"

*-------------------------------------------------------
*
* prepare qualifiers
*
*-------------------------------------------------------
SELECT quals
DELETE FOR area = m.area
PACK
COPY STRUCTURE TO (m.new + "quals")
SELECT 0
USE (m.new + "quals") ALIAS newquals
INDEX ON id TO (m.new + "quals")
SET INDEX TO (m.new + "quals")

*-------------------------------------------------------
*
* generate rule ==> qualifier cross-references
*
*-------------------------------------------------------
SELECT rule
SEEK m.area                                && set filters

DO WHILE area = m.area .AND. !EOF()     && process the rulebase
   * process rule input cross-references
   SELECT premise                          && premise clauses for this r
=> ule
   SEEK rule.premise
   DO WHILE clause = rule.premise .AND. !EOF()
      IF EMPTY( premise.op )
         SELECT fact                       && clause is a fact
         SEEK premise.fact
         DO WHILE clause = premise.fact .AND. !EOF()
            SELECT newquals
            SEEK fact.id
            IF !FOUND()
               APPEND BLANK                   && define new qualifier ob
=> ject
               REPLACE id WITH fact.id, OBJECT WITH fact.object
            ENDIF
            REPLACE area WITH m.area
            m.rules = rules
            m.s = ALLTRIM( STR( rule.rule ) )
            * check for duplicates
            i = AT( m.s, m.rules )
            IF i > 0
               IF VAL( SUBSTR( m.rules,i ) )  =  rule.rule
                  * skip duplicate entry
                  m.s = ""
               ENDIF
            ENDIF
            IF !EMPTY( m.s )
               m.rules = m.rules + IIF( !EMPTY( m.rules ), "|", "" )
=> + m.s
               REPLACE rules WITH m.rules
            ENDIF
            m.quals = rule.quals
            m.s = ALLTRIM( STR( fact.id ) )
            * check for duplicates
            i = AT(m.s, m.quals)
            IF i > 0
               IF VAL( SUBSTR( m.quals,i ) )  =  fact.id
                  * skip duplicate entry
                  m.s = ""
               ENDIF
            ENDIF
            IF !EMPTY( m.s )
               m.quals = m.quals + IIF( !EMPTY( m.quals ), "|", "" )
=> + m.s
               REPLACE rule.quals WITH m.quals
            ENDIF
            SELECT fact
            SKIP
         ENDDO
      ENDIF
      SELECT premise
      SKIP
   ENDDO

   * process rule output cross-references
   SELECT action                           && rule actions
   SEEK rule.action
   DO WHILE clause = rule.action .AND. !EOF()
      SELECT newquals
      SEEK action.id
      IF !FOUND()
         APPEND BLANK                      && define qualifier/goal
         REPLACE id WITH action.id, OBJECT WITH action.object
      ENDIF
      REPLACE area WITH m.area
      m.rules = ruleso
      m.s = ALLTRIM( STR( rule.rule ) )
      * check for duplicates
      i = AT(m.s, m.rules)
      IF i > 0
         IF VAL( SUBSTR( m.rules,i ) )  =  rule.rule
            * skip duplicate entry
            m.s = ""
         ENDIF
      ENDIF
      IF !EMPTY( m.s )
         m.rules = m.rules + IIF( !EMPTY( m.rules ), "|", "" ) + m.s
         REPLACE ruleso WITH m.rules
      ENDIF
      m.goals = rule.goals
      m.s = ALLTRIM( STR( action.id ) )
      * check for duplicates
      i = AT( m.s, m.goals )
      IF i > 0
         IF VAL( SUBSTR( m.goals,i ) )  =  action.id
            * skip duplicate entry
            m.s = ""
         ENDIF
      ENDIF
      IF !EMPTY( m.s )
         m.goals = m.goals + IIF( !EMPTY( m.goals ), "|", "" ) + m.s
         REPLACE rule.goals WITH m.goals
      ENDIF
      SELECT action
      SKIP
   ENDDO

   SELECT rule                             && select another rule
   SKIP
ENDDO

* copy qualifier cross-references
SELECT newquals
USE
SELECT quals
```

```
     APPEND FROM (m.new + "quals")
     DELETE FILE (m.new + "quals.dbf")
     DELETE FILE (m.new + "quals.idx")
     DELETE FILE (m.new + "quals.fpt")
     DELETE FILE (m.new + "area.dbf")
     DELETE FILE (m.new + "rule.dbf")
     DELETE FILE (m.new + "rule.dbt")
     DELETE FILE (m.new + "premise.dbf")
     DELETE FILE (m.new + "fact.dbf")
     DELETE FILE (m.new + "fact.dbt")
     DELETE FILE (m.new + "action.dbf")
     DELETE FILE (m.new + "action.dbt")
     DELETE FILE (m.new + "disease.dbf")
     DELETE FILE (m.new + "disease.dbt")
     DELETE FILE (m.new + "dict.dbf")
     DELETE FILE (m.new + "val.dbf")
     DELETE FILE (m.new + "enum.dbf")
     DELETE FILE (m.new + "goals.dbf")
     DELETE FILE (m.new + "display.dbf")
     DELETE FILE (m.new + "help.dbf")
     DELETE FILE (m.new + "help.dbt")

     SELECT area
     GOTO m.newrecno

     * close unuse database
     SELECT fact
     USE
     SELECT premise
     USE
     SELECT action
     USE
     SELECT rule
     USE

<——RETURN

 *:  EOF: KBLDR.act
```

```
*
*
*      08/09/94              ACTION.SPR              09:39:14
*
*
*      Author's Name
*
*      Copyright (c) 1994 Company Name
*      Address
*      City,     Zip
*
*      Description:
*      This program was automatically generated by GENSCRN.
*
*
*

DO CASE
CASE _WINDOWS


    #REGION 0
    REGIONAL m.currarea, m.talkstat, m.compstat

    IF SET("TALK") = "ON"
        SET TALK OFF
        m.talkstat = "ON"
    ELSE
        m.talkstat = "OFF"
    ENDIF
    m.compstat = SET("COMPATIBLE")
    SET COMPATIBLE FOXPLUS

    m.rborder = SET("READBORDER")
    SET readborder ON

    *
    *
    *                  Windows Window definitions
    *
    *
    *

    IF NOT WEXIST("w_act") ;
            OR UPPER(WTITLE("W_ACT")) == "W_ACT.PJX" ;
            OR UPPER(WTITLE("W_ACT")) == "W_ACT.SCX" ;
            OR UPPER(WTITLE("W_ACT")) == "W_ACT.MNX" ;
            OR UPPER(WTITLE("W_ACT")) == "W_ACT.PRG" ;
            OR UPPER(WTITLE("W_ACT")) == "W_ACT.FRX" ;
            OR UPPER(WTITLE("W_ACT")) == "W_ACT.QPR"
        DEFINE WINDOW w_act ;
            AT  0.000, 0.000  ;
            SIZE 14.500,57.250 ;
            TITLE "Action [THEN] Editor" ;
            FONT "Terminal", 8 ;
            FLOAT ;
            CLOSE ;
            SHADOW ;
            NOMINIMIZE ;
            COLOR RGB(,,,0,255,255)
        MOVE WINDOW w_act CENTER
    ENDIF


    *
    *
    *          ACTION/Windows Setup Code - SECTION 2
    *
    *
    *

    #REGION 1
    EXTERNAL ARRAY act
    PRIVATE mselect, mok
    mselect = SELECT()
    SELECT action
    IF EOF()
        = errmsg("No < THEN > Statment !!!",1)
<=======RETURN
    ENDIF


    *
    *
    *              ACTION/Windows Screen Layout
    *
    *
    *

    #REGION 1
    IF WVISIBLE("w_act")
        ACTIVATE WINDOW w_act SAME
    ELSE
        ACTIVATE WINDOW w_act NOSHOW
    ENDIF
    @ 3.000,47.500 GET mbuttons ;
        PICTURE "@*VN \<Edit;\<Quit" ;
        SIZE 3.000,7.500,1.083 ;
        DEFAULT 1 ;
        FONT "Terminal", 8 ;
        VALID _qsf0kox5v()
    @ 1.750,2.500 GET ma ;
        PICTURE "@&N" ;
        FROM act ;
        SIZE 10.500,43.375 ;
        DEFAULT 1 ;
        FONT "Terminal", 8 ;
        VALID _qsf0kox9d()

    IF NOT WVISIBLE("w_act")
        ACTIVATE WINDOW w_act
    ENDIF

    READ CYCLE MODAL

    RELEASE WINDOW w_act

    #REGION 0

    SET readborder &rborder

    IF m.talkstat = "ON"
       SET TALK ON
    ENDIF
    IF m.compstat = "ON"
       SET COMPATIBLE ON
    ENDIF

    *
    *
    *           ACTION/Windows Cleanup Code
    *
    *
    *

    #REGION 1
    SELECT (mselect)
    RETURN


CASE _DOS

    #REGION 0
    REGIONAL m.currarea, m.talkstat, m.compstat

    IF SET("TALK") = "ON"
       SET TALK OFF
       m.talkstat = "ON"
    ELSE
       m.talkstat = "OFF"
    ENDIF
    m.compstat = SET("COMPATIBLE")
    SET COMPATIBLE FOXPLUS

    *
    *
    *           MS-DOS Window definitions
    *
    *
    *

    IF NOT WEXIST("w_act") ;
          OR UPPER(WTITLE("W_ACT")) == "W_ACT.PJX" ;
          OR UPPER(WTITLE("W_ACT")) == "W_ACT.SCX" ;
          OR UPPER(WTITLE("W_ACT")) == "W_ACT.MNX" ;
          OR UPPER(WTITLE("W_ACT")) == "W_ACT.PRG" ;
          OR UPPER(WTITLE("W_ACT")) == "W_ACT.FRX" ;
          OR UPPER(WTITLE("W_ACT")) == "W_ACT.QPR"
       DEFINE WINDOW w_act ;
          FROM INT((SROW()-15)/2),INT((SCOL()-76)/2) ;
          TO INT((SROW()-15)/2)+14,INT((SCOL()-76)/2)+75 ;
          TITLE "Action Editor" ;
          FLOAT ;
          CLOSE ;
          SHADOW ;
          NOMINIMIZE ;
          COLOR SCHEME 1
    ENDIF


    *
    *
    *           ACTION/MS-DOS Setup Code - SECTION 2
    *
    *
    *

    #REGION 1
    EXTERNAL ARRAY act
    PRIVATE mselect, mok
    mselect = SELECT()
    SELECT action
    IF EOF()
       GOTO BOTTOM
       m.clause = IIF(RECCOUNT() = 0, 0, clause) + 1
       m.action = addnewact()
    ENDIF


    *
    *
    *           ACTION/MS-DOS Screen Layout
    *
    *
    *

    #REGION 1
    IF WVISIBLE("w_act")
       ACTIVATE WINDOW w_act SAME
    ELSE
       ACTIVATE WINDOW w_act NOSHOW
    ENDIF
    @ 5,64 GET mbuttons ;
       PICTURE "@*VT \<Delete;\<OK;\<Cancel" ;
       SIZE 1,8,1 ;
       DEFAULT 1 ;
       VALID _qsf0koxva()
    @ 1,64 GET mebutton ;
       PICTURE "@*VN \<Edit" ;
       SIZE 1,8,1 ;
       DEFAULT 1 ;
       WHEN ma > 0 ;
       VALID _qsf0koy5o()
    @ 3,64 GET minsbutton ;
       PICTURE "@*VN \<Add" ;
       SIZE 1,8,1 ;
```

```
         DEFAULT 1 ;
         VALID _qsf0koy8a()
      @ 1,1 GET ma ;
         PICTURE "@&N" ;
         FROM act ;
         SIZE 10,61 ;
         DEFAULT 1 ;
         VALID _qsf0koycp() ;
         COLOR SCHEME 2

      IF NOT WVISIBLE("w_act")
         ACTIVATE WINDOW w_act
      ENDIF

      READ CYCLE MODAL

      RELEASE WINDOW w_act

      #REGION 0
      IF m.talkstat = "ON"
         SET TALK ON
      ENDIF
      IF m.compstat = "ON"
         SET COMPATIBLE ON
      ENDIF


      *
      *
      *         ACTION/MS-DOS Cleanup Code
      *
      *
      *

      #REGION 1
      SELECT (mselect)
   <--RETURN



   ENDCASE


   *
   *
   *     _QSFOKOX5V           mbuttons VALID
   *
   *     Function Origin:
   *
   *     From Platform:       Windows
   *     From Screen:         ACTION,      Record Number:    2
   *     Variable:            mbuttons
   *     Called By:           VALID Clause
   *     Object Type:         Push Button
   *     Snippet Number:      1
   *
   *
   *
   FUNCTION _qsf0kox5v     &&  mbuttons VALID
   #REGION 1
   DO CASE
   CASE mbuttons = 1    && Edit
      IF ma > 0
         DO edact
      ENDIF
   CASE mbuttons = 2    && Quit
      CLEAR READ
   ENDCASE

   *
   *
   *     _QSFOKOX9D           ma VALID
   *
   *     Function Origin:
   *
   *     From Platform:       Windows
   *     From Screen:         ACTION,      Record Number:    3
   *     Variable:            ma
   *     Called By:           VALID Clause
   *     Object Type:         List
   *     Snippet Number:      2
   *
   *
   *
   FUNCTION _qsf0kox9d     &&  ma VALID
   #REGION 1
   DO edact

   *
   *
   *     _QSFOKOXVA           mbuttons VALID
   *
   *     Function Origin:
   *
   *     From Platform:       MS-DOS
   *     From Screen:         ACTION,      Record Number:    6
   *     Variable:            mbuttons
   *     Called By:           VALID Clause
   *     Object Type:         Push Button
   *     Snippet Number:      3
   *
   *
   *
   FUNCTION _qsf0koxva     &&  mbuttons VALID
   #REGION 1
   DO CASE
   CASE mbuttons = 1          && DELETE
      SEEK rule.action
      DO WHILE clause = rule.action
         DELETE
         SKIP
      ENDDO
      PACK
      mok = .F.
   CASE mbuttons = 2    && ok
      mok = .T.
   CASE mbuttons = 3    && cancel
      mok = .F.
   ENDCASE

   *
   *
   *     _QSFOKOY5O           mebutton VALID
   *
   *     Function Origin:
   *
   *     From Platform:       MS-DOS
```

```
*       From Screen:       ACTION,     Record Number:   7
*       Variable:          mebutton
*       Called By:         VALID Clause
*       Object Type:       Push Button
*       Snippet Number:    4
*
*
*
FUNCTION _qsf0koy5o     &&  mebutton VALID
#REGION 1
DO edact

*
*
*       _QSFOKOY8A          minsbutton VALID
*
*       Function Origin:
*
*       From Platform:     MS-DOS
*       From Screen:       ACTION,     Record Number:   8
*       Variable:          minsbutton
*       Called By:         VALID Clause
*       Object Type:       Push Button
*       Snippet Number:    5
*
*
*
FUNCTION _qsf0koy8a     &&  minsbutton VALID
#REGION 1
PRIVATE m.sel
m.sel = SELECT()
m.action = rule.action
SELECT action
APPEND BLANK
REPLACE clause WITH m.action
DO clause.spr WITH m.action
┌IF EMPTY(op) AND EMPTY(OBJECT) AND EMPTY(id)
│   DELETE
│   PACK
└ENDIF
SELECT (m.sel)
DO setact
*show get mbbutton enable
*show gets



*
*
*       _QSFOKOYCP          ma VALID
*
*       Function Origin:
*
*       From Platform:     MS-DOS
*       From Screen:       ACTION,     Record Number:   9
*       Variable:          ma
*       Called By:         VALID Clause
*       Object Type:       List
*       Snippet Number:    6
*
*
*
FUNCTION _qsf0koycp     &&  ma VALID
#REGION 1
DO edact

```

```

*
*
*       ACTION/MS-DOS Supporting Procedures and Functions
*
*
*

#REGION 1
PROCEDURE edact
EXTERNAL ARRAY actr
SET TOPIC TO "EDIT"
SELECT "ACTION"
┌IF actr[ma] > 0
│   GOTO actr[ma]
└ENDIF
m.clause = rule.action
DO clause.spr
DO setact
SHOW GETS
mtopic = ALIAS()
SET TOPIC TO &mtopic
<──RETURN


FUNCTION addnewact
PRIVATE m.sel
m.sel = SELECT()
SELECT action
GOTO BOTTOM
m.action = IIF(EOF(),0,clause) + 1
DO clause.spr WITH m.action
SEEK m.action
┌IF FOUND()
│   SELECT rule
│   REPLACE action WITH m.action
└ENDIF
SELECT (m.sel)
DO setact
<──RETURN
*:  EOF: ACTION.ac1
```

```
*
*
*        08/09/94          ACTELSE.SPR          09:39:18
*
*
*        Author's Name
*
*        Copyright (c) 1994 Company Name
*        Address
*        City,     Zip
*
*        Description:
*        This program was automatically generated by GENSCRN.
*
*

DO CASE
CASE _WINDOWS


   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
   ENDIF
   m.compstat = SET("COMPATIBLE")
   SET COMPATIBLE FOXPLUS

   m.rborder = SET("READBORDER")
   SET readborder ON

   *
   *
   *                 Windows Window definitions
   *
   *
   *

   IF NOT WEXIST("w_act") ;
         OR UPPER(WTITLE("W_ACT")) == "W_ACT.PJX" ;
         OR UPPER(WTITLE("W_ACT")) == "W_ACT.SCX" ;
         OR UPPER(WTITLE("W_ACT")) == "W_ACT.MNX" ;
         OR UPPER(WTITLE("W_ACT")) == "W_ACT.PRG" ;
         OR UPPER(WTITLE("W_ACT")) == "W_ACT.FRX" ;
         OR UPPER(WTITLE("W_ACT")) == "W_ACT.QPR"
      DEFINE WINDOW w_act ;
         AT  0.000, 0.000  ;
         SIZE 14.333,57.000 ;
         TITLE "Action [ELSE] se) Editor" ;
         FONT "Terminal", 8 ;
         FLOAT ;
         CLOSE ;
         SHADOW ;
         NOMINIMIZE ;
         COLOR RGB(,,,0,255,255)
      MOVE WINDOW w_act CENTER
   ENDIF


   *
   *
   *        ACTELSE/Windows Setup Code - SECTION 2
   *
   *
   *

   #REGION 1
   EXTERNAL ARRAY actelse
   PRIVATE mselect, mok
   mselect = SELECT()
   SELECT action
   IF EOF()
      =errmsg("No <Else> statement !!!",1)
   ENDIF


   *
   *
   *        ACTELSE/Windows Screen Layout
   *
   *
   *

   #REGION 1
   IF WVISIBLE("w_act")
      ACTIVATE WINDOW w_act SAME
   ELSE
      ACTIVATE WINDOW w_act NOSHOW
   ENDIF
   @ 4.333,47.750 GET mbuttons ;
      PICTURE "@*VN \<Edit;\<Quit" ;
      SIZE 2.583,7.500,1.083 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kp07d()
   @ 2.000,2.375 GET me ;
      PICTURE "@&N" ;
      FROM actelse ;
      SIZE 10.500,43.250 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kp0ar()

   IF NOT WVISIBLE("w_act")
      ACTIVATE WINDOW w_act
   ENDIF

   READ CYCLE MODAL

   RELEASE WINDOW w_act

   #REGION 0
```

```
SET readborder &rborder

IF m.talkstat = "ON"
   SET TALK ON
ENDIF
IF m.compstat = "ON"
   SET COMPATIBLE ON
ENDIF


*
*
*          ACTELSE/Windows Cleanup Code
*
*
*

#REGION 1
SELECT (mselect)
RETURN


CASE _DOS


#REGION 0
REGIONAL m.currarea, m.talkstat, m.compstat

IF SET("TALK") = "ON"
   SET TALK OFF
   m.talkstat = "ON"
ELSE
   m.talkstat = "OFF"
ENDIF
m.compstat = SET("COMPATIBLE")
SET COMPATIBLE FOXPLUS

*
*
*          MS-DOS Window definitions
*
*
*

IF NOT WEXIST("w_act") ;
      OR UPPER(WTITLE("W_ACT")) == "W_ACT.PJX" ;
      OR UPPER(WTITLE("W_ACT")) == "W_ACT.SCX" ;
      OR UPPER(WTITLE("W_ACT")) == "W_ACT.MNX" ;
      OR UPPER(WTITLE("W_ACT")) == "W_ACT.PRG" ;
      OR UPPER(WTITLE("W_ACT")) == "W_ACT.FRX" ;
      OR UPPER(WTITLE("W_ACT")) == "W_ACT.QPR"
   DEFINE WINDOW w_act ;
      FROM INT((SROW()-15)/2),INT((SCOL()-76)/2) ;
      TO INT((SROW()-15)/2)+14,INT((SCOL()-76)/2)+75 ;
      TITLE "Action (Else) Editor" ;
      FLOAT ;
      CLOSE ;
      SHADOW ;
      NOMINIMIZE ;
      COLOR SCHEME 1
ENDIF


*
*
*          ACTELSE/MS-DOS Setup Code - SECTION 2
*
*
*


#REGION 1
EXTERNAL ARRAY actelse
PRIVATE mselect, mok
mselect = SELECT()
SELECT action
IF EOF()
   GOTO BOTTOM
   m.clause = IIF(RECCOUNT() = 0, 0, clause) + 1
   m.action = addnewelse()
ENDIF



*
*
*          ACTELSE/MS-DOS Screen Layout
*
*
*


#REGION 1
IF WVISIBLE("w_act")
   ACTIVATE WINDOW w_act SAME
ELSE
   ACTIVATE WINDOW w_act NOSHOW
ENDIF
@ 5,64 GET mbuttons ;
   PICTURE "@*VT \<Delete;\<OK;\<Cancel" ;
   SIZE 1,8,1 ;
   DEFAULT 1 ;
   VALID _qsf0kp0s3()
@ 1,64 GET mebutton ;
   PICTURE "@*VN \<Edit" ;
   SIZE 1,8,1 ;
   DEFAULT 1 ;
   WHEN me > 0 ;
   VALID _qsf0kp0w3()
@ 3,64 GET minsbutton ;
   PICTURE "@*VN \<Add" ;
   SIZE 1,8,1 ;
   DEFAULT 1 ;
```

```
      VALID _qsf0kp0yp()
   @ 1,1 GET me ;
      PICTURE "@&N" ;
      FROM actelse ;
      SIZE 10,61 ;
      DEFAULT 1 ;
      VALID _qsf0kp13m() ;
      COLOR SCHEME 2

   IF NOT WVISIBLE("w_act")
      ACTIVATE WINDOW w_act
   ENDIF

   READ CYCLE MODAL

   RELEASE WINDOW w_act

   #REGION 0
   IF m.talkstat = "ON"
      SET TALK ON
   ENDIF
   IF m.compstat = "ON"
      SET COMPATIBLE ON
   ENDIF


   *
   *
   *           ACTELSE/MS-DOS Cleanup Code
   *
   *
   *

   #REGION 1
   SELECT (mselect)
<--RETURN


ENDCASE


 *
 *
 *       _QSFOKP07D           mbuttons VALID
 *
 *       Function Origin:
 *
 *       From Platform:       Windows
 *       From Screen:         ACTELSE,     Record Number:    2
 *       Variable:            mbuttons
 *       Called By:           VALID Clause
 *       Object Type:         Push Button
 *       Snippet Number:      1
 *
 *
 *
 FUNCTION _qsf0kp07d     &&  mbuttons VALID
 #REGION 1
 DO CASE
 CASE mbuttons = 1   && Edit
   IF me > 0
      DO edelse
   ENDIF
 CASE mbuttons = 2   && ok
    CLEAR READ
 ENDCASE

 *
 *
 *       _QSFOKP0AR           me VALID
 *
 *       Function Origin:
 *
 *       From Platform:       Windows
 *       From Screen:         ACTELSE,     Record Number:    3
 *       Variable:            me
 *       Called By:           VALID Clause
 *       Object Type:         List
 *       Snippet Number:      2
 *
 *
 *
 FUNCTION _qsf0kp0ar     &&  me VALID
 #REGION 1
 DO edelse


 *
 *
 *       _QSFOKP0S3           mbuttons VALID
 *
 *       Function Origin:
 *
 *       From Platform:       MS-DOS
 *       From Screen:         ACTELSE,     Record Number:    6
 *       Variable:            mbuttons
 *       Called By:           VALID Clause
 *       Object Type:         Push Button
 *       Snippet Number:      3
 *
 *
 *
 FUNCTION _qsf0kp0s3     &&  mbuttons VALID
 #REGION 1
 DO CASE
 CASE mbuttons = 1          && DELETE
    SEEK rule.else
    DO WHILE clause = rule.else
       DELETE
       SKIP
    ENDDO
    PACK
    mok = .F.
 CASE mbuttons = 2   && ok
    mok = .T.
 CASE mbuttons = 3   && cancel
    mok = .F.
 ENDCASE

 *
 *
 *       _QSFOKP0W3           mebutton VALID
 *
 *       Function Origin:
 *
 *       From Platform:       MS-DOS
 *       From Screen:         ACTELSE,     Record Number:    7
*         Variable:              mebutton
*         Called By:             VALID Clause
*         Object Type:           Push Button
*         Snippet Number:        4
*
*
*
FUNCTION _qsf0kp0w3     &&  mebutton VALID
#REGION 1
DO edelse

*
*
*          _QSFOKPOYP             minsbutton VALID
*
*          Function Origin:
*
*          From Platform:         MS-DOS
*          From Screen:           ACTELSE,      Record Number:    8
*          Variable:              minsbutton
*          Called By:             VALID Clause
*          Object Type:           Push Button
*          Snippet Number:        5
*
*
*
FUNCTION _qsf0kp0yp     &&  minsbutton VALID
#REGION 1
PRIVATE m.sel
m.sel = SELECT()
SELECT action
IF rule.else = 0
   GOTO BOTTOM
   m.clause = IIF(RECCOUNT() = 0, 0, clause) + 1
   = addnewelse()
ELSE
   m.action = rule.else
   APPEND BLANK
   REPLACE clause WITH m.action
   DO clause.spr WITH m.action
   IF EMPTY(op) AND EMPTY(OBJECT) AND EMPTY(id)
      DELETE
      PACK
   ENDIF
ENDIF
SELECT (m.sel)
DO setelse
SHOW GETS



*
*
*          _QSFOKP13M             me VALID
*
*          Function Origin:
*
*          From Platform:         MS-DOS
*          From Screen:           ACTELSE,      Record Number:    9
*          Variable:              me
*          Called By:             VALID Clause
*          Object Type:           List
*          Snippet Number:        6
*
*
*
FUNCTION _qsf0kp13m     &&  me VALID
#REGION 1
DO edelse



*
*
*          ACTELSE/MS-DOS Supporting Procedures and Functions
*
*
*

#REGION 1

*
*
*                 ACTELSE Procedure EDELSE
*
*
*


PROCEDURE edelse
DO CASE
CASE _DOS
   EXTERNAL ARRAY actelser
   SET TOPIC TO "EDIT"
   SELECT "ACTION"
   IF actelser[me] > 0
      GOTO actelser[me]
   ENDIF
   m.clause = rule.else
   DO clause.spr
   DO setelse
   SHOW GETS
   mtopic = ALIAS()
   SET TOPIC TO &mtopic
<--RETURN

CASE _WINDOWS
   EXTERNAL ARRAY actelser
   SET TOPIC TO "EDIT"
   SELECT "ACTION"
   IF actelser[me] > 0
      GOTO actelser[me]
   ENDIF
   m.clause = rule.else
   DO clause.spr
   DO setelse
   SHOW GETS
   mtopic = ALIAS()
   SET TOPIC TO &mtopic
<--RETURN

ENDCASE

*
*
*                 ACTELSE Procedure ADDELSE
*
*
*


PROCEDURE addelse
```

```
┌DO CASE
├CASE _DOS
│   EXTERNAL ARRAY actelser
│   SET TOPIC TO "ADD"
│   SELECT "ACTION"
│   SCATTER MEMVAR BLANK
│   m.clause = rule.else
│   DO clause.spr
│   DO setelse
│   SHOW GETS
│   mtopic = ALIAS()
│   SET TOPIC TO &mtopic
<───RETURN
│
├CASE _WINDOWS
│   EXTERNAL ARRAY actelser
│   SET TOPIC TO "ADD"
│   SELECT "ACTION"
│   SCATTER MEMVAR BLANK
│   m.clause = rule.else
│   DO clause.spr
│   DO setelse
│   SHOW GETS
│   mtopic = ALIAS()
│   SET TOPIC TO &mtopic
<───RETURN
│
│
└ENDCASE

  *
  *
  *          ACTELSE Function ADDNEWELSE
  *
  *
  *

  FUNCTION addnewelse
  PRIVATE m.sel
  m.sel = SELECT()
  SELECT action
  m.action = m.clause
  DO clause.spr WITH m.action
  SEEK m.action
┌IF FOUND()
│   SELECT rule
│   REPLACE ELSE WITH m.action
└ENDIF
  SELECT (m.sel)
  DO setelse
<──RETURN
*:  EOF: ACTELSE.ac1
```

```
*
*
*      08/09/94            CLAUSE.SPR              09:39:22
*
*
*
*      Author's Name
*
*      Copyright (c) 1994 Company Name
*      Address
*      City,     Zip
*
*      Description:
*      This program was automatically generated by GENSCRN.
*
*

PARAMETERS m.clause
DO CASE
CASE _WINDOWS


    *
    *
    *           CLAUSE/Windows Setup Code - SECTION 1
    *
    *
    *

    #REGION 1
    PRIVATE mobj, mselect, mtype, mok, mrecno, u
    PRIVATE m.clause, m.op, m.object, m.id, m.tag, m.val, m.text
    DIMENSION mrecno[2]
    m.adding = .F.
    mselect = SELECT()
    *select fact
    m.clause = IIF(PARAMETER() = 0, clause, m.clause)
    IF m.clause <> clause
       SEEK m.clause
       IF !FOUND()
          m.adding = .T.
          m.op = " "
          m.object = ""
          mtype = ""
          = setobject(@m.object, @mtype)
          m.id = 0
          m.val = " "
          m.tag = "F"
       ENDIF
    ELSE
       SCATTER MEMVAR
       IF m.tag = "T"
          m.val = TEXT
       ENDIF
       IF EMPTY(m.object)
          mtype = ""
          =setobject(@m.object, @mtype)
       ENDIF
       IF m.object = "D"
          SELECT disease
          SEEK m.id
          mobj = name
          mtype = "Goal"
       ELSE
          SELECT dict
          SEEK m.id
          mobj = name
          mtype = "Qualifier"
       ENDIF
    ENDIF
    mok = .F.
    SELECT dict
    mrecno[1] = IIF ( EOF(), 0, RECNO() )
    SELECT disease
    mrecno[2] = IIF ( EOF(), 0, RECNO() )
    SELECT (mselect)


    #REGION 0
    REGIONAL m.currarea, m.talkstat, m.compstat

    IF SET("TALK") = "ON"
       SET TALK OFF
       m.talkstat = "ON"
    ELSE
       m.talkstat = "OFF"
    ENDIF
    m.compstat = SET("COMPATIBLE")
    SET COMPATIBLE FOXPLUS

    m.rborder = SET("READBORDER")
    SET readborder ON

    *
    *
    *           Windows Window definitions
    *
    *
    *

    IF NOT WEXIST("w_clause") ;
          OR UPPER(WTITLE("W_CLAUSE")) == "W_CLAUSE.PJX" ;
          OR UPPER(WTITLE("W_CLAUSE")) == "W_CLAUSE.SCX" ;
          OR UPPER(WTITLE("W_CLAUSE")) == "W_CLAUSE.MNX" ;
          OR UPPER(WTITLE("W_CLAUSE")) == "W_CLAUSE.PRG" ;
          OR UPPER(WTITLE("W_CLAUSE")) == "W_CLAUSE.FRX" ;
          OR UPPER(WTITLE("W_CLAUSE")) == "W_CLAUSE.QPR"
       DEFINE WINDOW w_clause ;
          AT  0.000, 0.000  ;
          SIZE 23.167,55.375 ;
          TITLE "Clause Editor" ;
          FONT "Terminal", 8 ;
          FLOAT ;
          CLOSE ;
          SHADOW ;
          NOMINIMIZE ;
          COLOR RGB(,,,128,128,0)
       MOVE WINDOW w_clause CENTER
    ENDIF


    *
```

```
     *
     *           CLAUSE/Windows Screen Layout
     *
     *
     *

     #REGION 1
     IF WVISIBLE("w_clause")
        ACTIVATE WINDOW w_clause SAME
     ELSE
        ACTIVATE WINDOW w_clause NOSHOW
     ENDIF
     @ 11.250,2.500 GET minval ;
        PICTURE "@*IVN " ;
        SIZE 1.167,9.000,1.083 ;
        DEFAULT 0 ;
        FONT "Terminal", 8 ;
        VALID _qsf0kp3sz()
     @ 4.583,2.625 GET minvbutton ;
        PICTURE "@*IVN " ;
        SIZE 1.250,9.000,1.083 ;
        DEFAULT 0 ;
        FONT "Terminal", 8 ;
        VALID _qsf0kp3x2()
     @ 11.333,2.625 SAY "<Value>"  ;
        FONT "Terminal", 8
     @ 4.667,2.625 SAY "<Object>"  ;
        FONT "Terminal", 8
     @ 1.083,2.625 SAY "Type"  ;
        FONT "Terminal", 8
     @ 8.167,2.625 SAY "Operator"  ;
        FONT "Terminal", 8
     @ 1.000,43.625 GET mbuttons ;
        PICTURE "@*VT \<OK;\<Cancel" ;
        SIZE 1.917,7.500,1.083 ;
        DEFAULT 1 ;
        FONT "Terminal", 8 ;
        VALID _qsf0kp435()
     @ 1.083,12.625 GET mtype ;
        PICTURE "@^ Qualifier;Goal" ;
        SIZE 1.500,23.000 ;
        DEFAULT "Qualifier" ;
        FONT "Terminal", 8 ;
        VALID _qsf0kp472()
     @ 4.583,12.625 GET mobj ;
        SIZE 1.000,23.125 ;
        DEFAULT " " ;
        FONT "Terminal", 8 ;
        DISABLE
     @ 7.917,12.500 GET m.op ;
        PICTURE "@^ <;<=;==;>=;>;!=;=" ;
        SIZE 1.500,5.000 ;
        DEFAULT "<" ;
        FONT "Terminal", 8
     @ 12.917,2.750 EDIT m.val ;
        SIZE 9.167,48.625,0.000 ;
        DEFAULT " " ;
        FONT "Terminal", 8 ;
        SCROLL ;
        DISABLE

     IF NOT WVISIBLE("w_clause")
        ACTIVATE WINDOW w_clause
     ENDIF

     READ CYCLE MODAL

     RELEASE WINDOW w_clause

     #REGION 0

     SET readborder &rborder

     IF m.talkstat = "ON"
        SET TALK ON
     ENDIF
     IF m.compstat = "ON"
        SET COMPATIBLE ON
     ENDIF


     *
     *
     *           CLAUSE/Windows Cleanup Code
     *
     *
     *

     #REGION 1
     IF mok
        IF m.adding
           APPEND BLANK
           REPLACE clause WITH m.clause
        ENDIF
        m.val = ALLTRIM(m.val)
        GATHER MEMVAR
        IF m.tag = "T"  .OR. LEN(m.val) > LEN(VAL)
           REPLACE TEXT WITH m.val, TAG WITH "T",VAL WITH " "
        ENDIF
        IF LEFT(mtype,1) = "G"
           m.temp = rule.goals
           m.search = insnewid(@m.temp,m.id)
           IF !m.search
              m.cur = SELECT()
              SELECT rule
              REPLACE goals WITH m.temp
              SELECT (m.cur)
              RELEASE meme m.cur
           ENDIF
        ELSE
           m.temp = rule.quals
           m.search = insnewid(@m.temp,m.id)
           IF !m.search
              m.cur = SELECT()
              SELECT rule
              REPLACE quals WITH m.temp
              SELECT (m.cur)
              RELEASE meme m.cur
           ENDIF
        ENDIF
     ELSE
       IF mrecno[1] > 0
```

```
            SELECT dict
            GOTO mrecno[1]
         ENDIF
         IF mrecno[2] > 0
            SELECT disease
            GOTO mrecno[2]
         ENDIF
      ENDIF
      SELECT (mselect)
      RETURN


   CASE _DOS


      *      CLAUSE/MS-DOS Setup Code - SECTION 1
      *


      #REGION 1
      PRIVATE mobj, mselect, mtype, mok, mrecno, u
      PRIVATE m.clause, m.op, m.object, m.id, m.tag, m.val, m.text
      DIMENSION mrecno[2]
      m.adding = .F.
      mselect = SELECT()
      *select fact
      m.clause = IIF(PARAMETER() = 0, clause, m.clause)
      IF m.clause <> clause
         SEEK m.clause
         IF !FOUND()
            m.adding = .T.
            m.op = " "
            m.object = ""
            mtype = ""
            = setobject(@m.object, @mtype)
            m.id = 0
            m.val = " "
            m.tag = "F"
         ENDIF
      ELSE
         SCATTER MEMVAR
         IF m.tag = "T"
            m.val = TEXT
         ENDIF
         IF EMPTY(m.object)
            mtype = ""
            =setobject(@m.object, @mtype)
         ENDIF
         IF m.object = "D"
            SELECT disease
            SEEK m.id
            mobj = name
            mtype = "Goal"
         ELSE
            SELECT dict
            SEEK m.id
            mobj = name
            mtype = "Qualifier"
         ENDIF
      ENDIF
      mok = .F.
      SELECT dict
      mrecno[1] = IIF ( EOF(), 0, RECNO() )
      SELECT disease
      mrecno[2] = IIF ( EOF(), 0, RECNO() )
      SELECT (mselect)


      #REGION 0
      REGIONAL m.currarea, m.talkstat, m.compstat

      IF SET("TALK") = "ON"
         SET TALK OFF
         m.talkstat = "ON"
      ELSE
         m.talkstat = "OFF"
      ENDIF
      m.compstat = SET("COMPATIBLE")
      SET COMPATIBLE FOXPLUS

      *      MS-DOS Window definitions
      *


      IF NOT WEXIST("w_clause") ;
            OR UPPER(WTITLE("W_CLAUSE")) == "W_CLAUSE.PJX" ;
            OR UPPER(WTITLE("W_CLAUSE")) == "W_CLAUSE.SCX" ;
            OR UPPER(WTITLE("W_CLAUSE")) == "W_CLAUSE.MNX" ;
            OR UPPER(WTITLE("W_CLAUSE")) == "W_CLAUSE.PRG" ;
            OR UPPER(WTITLE("W_CLAUSE")) == "W_CLAUSE.FRX" ;
            OR UPPER(WTITLE("W_CLAUSE")) == "W_CLAUSE.QPR"
         DEFINE WINDOW w_clause ;
            FROM INT((SROW()-20)/2),INT((SCOL()-68)/2) ;
            TO INT((SROW()-20)/2)+19,INT((SCOL()-68)/2)+67 ;
            TITLE "Clause Editor" ;
            FLOAT ;
            CLOSE ;
            SHADOW ;
            NOMINIMIZE ;
            COLOR SCHEME 1
      ENDIF


      *      CLAUSE/MS-DOS Screen Layout
      *

      #REGION 1
```

```
IF WVISIBLE("w_clause")
   ACTIVATE WINDOW w_clause SAME
ELSE
   ACTIVATE WINDOW w_clause NOSHOW
ENDIF
 @ 10,1 SAY "<Value>" ;
    SIZE 1,7, 0
 @ 4,1 SAY "<Object>" ;
    SIZE 1,8, 0
 @ 11,1 EDIT m.val ;
    SIZE 7,64,0 ;
    DEFAULT " " ;
    SCROLL ;
    DISABLE
 @ 4,10 GET mobj ;
    SIZE 1,37 ;
    DEFAULT " " ;
    DISABLE
 @ 1,1 SAY "Type" ;
    SIZE 1,4, 0
 @ 6,10 GET m.op ;
    PICTURE "@^ <;<=;==;>=;>;!=;=" ;
    SIZE 3,6 ;
    DEFAULT "<" ;
    COLOR SCHEME 1, 2
 @ 0,10 GET mtype ;
    PICTURE "@^ Qualifier;Goal" ;
    SIZE 3,11 ;
    DEFAULT "Qualifier" ;
    VALID _qsf0kp5hs() ;
    COLOR SCHEME 1, 2
 @ 10,1 GET minval ;
    PICTURE "@*IVN " ;
    SIZE 1,7,1 ;
    DEFAULT 0 ;
    VALID _qsf0kp5lz()
 @ 4,1 GET minvbutton ;
    PICTURE "@*IVN " ;
    SIZE 1,8,1 ;
    DEFAULT 0 ;
    VALID _qsf0kp5q1()
 @ 1,53 GET mbuttons ;
    PICTURE "@*VT \<OK;\<Cancel" ;
    SIZE 1,8,1 ;
    DEFAULT 1 ;
    VALID _qsf0kp5uc()
 @ 7,1 SAY "Operator" ;
    SIZE 1,8, 0

IF NOT WVISIBLE("w_clause")
   ACTIVATE WINDOW w_clause
ENDIF

 READ CYCLE MODAL

 RELEASE WINDOW w_clause

 #REGION 0
IF m.talkstat = "ON"
   SET TALK ON
ENDIF
IF m.compstat = "ON"
   SET COMPATIBLE ON
ENDIF


      *
      *      CLAUSE/MS-DOS Cleanup Code
      *
      *
      *
      *

      #REGION 1
     IF mok
       IF m.adding
          APPEND BLANK
          REPLACE clause WITH m.clause
       ENDIF
       m.val = ALLTRIM(m.val)
       GATHER MEMVAR
       IF m.tag = "T"  .OR. LEN(m.val) > LEN(VAL)
          REPLACE TEXT WITH m.val, TAG WITH "T",VAL WITH " "
       ENDIF
       IF LEFT(mtype,1) = "G"
          m.temp = rule.goals
          m.search = insnewid(@m.temp,m.id)
         IF !m.search
             m.cur = SELECT()
             SELECT rule
             REPLACE goals WITH m.temp
             SELECT (m.cur)
             RELEASE meme m.cur
         ENDIF
       ELSE
          m.temp = rule.quals
          m.search = insnewid(@m.temp,m.id)
         IF !m.search
             m.cur = SELECT()
             SELECT rule
             REPLACE quals WITH m.temp
             SELECT (m.cur)
             RELEASE meme m.cur
         ENDIF
       ENDIF
     ELSE
       IF mrecno[1] > 0
          SELECT dict
          GOTO mrecno[1]
       ENDIF
       IF mrecno[2] > 0
          SELECT disease
          GOTO mrecno[2]
       ENDIF
     ENDIF
      SELECT (mselect)
  <---RETURN

    ENDCASE


     *
     *
     *    _QSF0KP3SZ          minval VALID
     *
     *    Function Origin:
```

```
*
*          From Platform:       Windows
*          From Screen:         CLAUSE,      Record Number:    2
*          Variable:            minval
*          Called By:           VALID Clause
*          Snippet Number:      1
*
*
*
FUNCTION _qsf0kp3sz     &&  minval VALID
#REGION 1
DO CASE
CASE m.object = "S"
   DO CASE
   CASE dict.datatype = "N" .OR. dict.datatype = "C"
      SHOW GET m.val ENABLE
       CUROBJ = OBJNUM(m.val)
   CASE dict.datatype = "L" .OR. dict.datatype = "E" .OR. dict.dataty
=> pe = "M"
      SELECT enum
      BROWSE FOR id = dict.id NOEDIT
      m.val = ALLTRIM(enumerate)
      SELECT (mselect)
   ENDCASE
CASE m.object = "D"
   SHOW GET m.val ENABLE
    _CUROBJ = OBJNUM(m.val)
ENDCASE
SHOW GETS


*
*
*          _QSFOKP3X2           minvbutton VALID
*
*          Function Origin:
*
*          From Platform:       Windows
*          From Screen:         CLAUSE,      Record Number:    3
*          Variable:            minvbutton
*          Called By:           VALID Clause
*          Snippet Number:      2
*
*
*
FUNCTION _qsf0kp3x2     &&  minvbutton VALID
#REGION 1
PRIVATE msel
msel = SELECT()
DO CASE
CASE m.object = "S"
   SELECT dict
   IF datatype = "N" .OR. datatype = "C"
      SHOW GET m.val ENABLE
   ELSE
      SHOW GET m.val DISABLE
   ENDIF
CASE m.object = "D"
   SELECT disease
   SHOW GET m.val ENABLE
ENDCASE
BROWSE NOEDIT
m.id = id
mobj = name
SELECT (msel)
SHOW GETS


*
*
*          _QSFOKP435           mbuttons VALID
*
*          Function Origin:
*
*          From Platform:       Windows
*          From Screen:         CLAUSE,      Record Number:    8
*          Variable:            mbuttons
*          Called By:           VALID Clause
*          Object Type:         Push Button
*          Snippet Number:      3
*
*
*
FUNCTION _qsf0kp435     &&  mbuttons VALID
#REGION 1
DO CASE
   *case mbuttons = 1   && delete
   *   m.sure = yesno("Sure you want to delete?","YES","NO")
   *   IF m.sure
   *      delete
   *      pack
   *   ENDIF
   *   m.id = 0
   *   m.op = " "
   *   m.val = " "
   *   mok = .f.
CASE mbuttons = 1   && ok
   mok = .T.
CASE mbuttons = 2   && cancel
   mok = .F.
ENDCASE

*
*
*          _QSFOKP472           mtype VALID
*
*          Function Origin:
*
*          From Platform:       Windows
*          From Screen:         CLAUSE,      Record Number:    9
*          Variable:            mtype
*          Called By:           VALID Clause
*          Object Type:         Popup
*          Snippet Number:      4
*
*
*
FUNCTION _qsf0kp472     &&  mtype VALID
#REGION 1
m.object = LEFT( mtype, 1 )
m.object = CHRTRAN( m.object,'GQ','DS' )
IF m.object = "S"
   SELECT dict
   IF datatype = "N" .OR. datatype = "C"
      SHOW GET m.val ENABLE
   ELSE
      SHOW GET m.val DISABLE
   ENDIF
ELSE
   SELECT disease
   SHOW GET m.val ENABLE
ENDIF
```

```
m.id = id
mobj = name
SELECT (mselect)
SHOW GETS


*
*    ┌──────────────────────────────────────────────────────────┐
*    │ _QSFOKP5HS           mtype VALID                         │
*    │                                                          │
*    │ Function Origin:                                         │
*    │                                                          │
*    │ From Platform:       MS-DOS                              │
*    │ From Screen:         CLAUSE,      Record Number:   21    │
*    │ Variable:            mtype                               │
*    │ Called By:           VALID Clause                        │
*    │ Object Type:         Popup                               │
*    │ Snippet Number:      5                                   │
*    │                                                          │
*    └──────────────────────────────────────────────────────────┘
*
FUNCTION _qsf0kp5hs     &&  mtype VALID
#REGION 1
m.object = LEFT( mtype, 1 )
m.object = CHRTRAN( m.object,'GQ','DS' )
┌IF m.object = "S"
│   SELECT dict
│  ┌IF datatype = "N" .OR. datatype = "C"
│  │   SHOW GET m.val ENABLE
│  ├ELSE
│  │   SHOW GET m.val DISABLE
│  └ENDIF
├ELSE
│   SELECT disease
│   SHOW GET m.val ENABLE
└ENDIF
m.id = id
mobj = name
SELECT (mselect)
SHOW GETS


*
*    ┌──────────────────────────────────────────────────────────┐
*    │ _QSFOKP5LZ           minval VALID                        │
*    │                                                          │
*    │ Function Origin:                                         │
*    │                                                          │
*    │ From Platform:       MS-DOS                              │
*    │ From Screen:         CLAUSE,      Record Number:   22    │
*    │ Variable:            minval                              │
*    │ Called By:           VALID Clause                        │
*    │ Snippet Number:      6                                   │
*    │                                                          │
*    └──────────────────────────────────────────────────────────┘
*
FUNCTION _qsf0kp5lz     &&  minval VALID
#REGION 1
┌DO CASE
├CASE m.object = "S"
│  ┌DO CASE
│  ├CASE dict.datatype = "N" .OR. dict.datatype = "C"
│  │    SHOW GET m.val ENABLE
│  │     CUROBJ = OBJNUM(m.val)
│  ├CASE dict.datatype = "L" .OR. dict.datatype = "E" .OR. dict.dataty
=> pe = "M"
│  │    SELECT enum
│  │    BROWSE FOR id = dict.id NOEDIT
│  │    m.val = ALLTRIM(enumerate)
│  │    SELECT (mselect)
│  └ENDCASE
├CASE m.object = "D"
│   SHOW GET m.val ENABLE
│    CUROBJ = OBJNUM(m.val)
└ENDCASE
SHOW GETS


*
*    ┌──────────────────────────────────────────────────────────┐
*    │ _QSFOKP5Q1           minvbutton VALID                    │
*    │                                                          │
*    │ Function Origin:                                         │
*    │                                                          │
*    │ From Platform:       MS-DOS                              │
*    │ From Screen:         CLAUSE,      Record Number:   23    │
*    │ Variable:            minvbutton                          │
*    │ Called By:           VALID Clause                        │
*    │ Snippet Number:      7                                   │
*    │                                                          │
*    └──────────────────────────────────────────────────────────┘
*
FUNCTION _qsf0kp5q1     &&  minvbutton VALID
#REGION 1
PRIVATE msel
msel = SELECT()
┌DO CASE
├CASE m.object = "S"
│   SELECT dict
│  ┌IF datatype = "N" .OR. datatype = "C"
│  │   SHOW GET m.val ENABLE
│  ├ELSE
│  │   SHOW GET m.val DISABLE
│  └ENDIF
├CASE m.object = "D"
│   SELECT disease
│   SHOW GET m.val ENABLE
└ENDCASE
BROWSE NOEDIT
m.id = id
mobj = name
SELECT (msel)
SHOW GETS


*
*    ┌──────────────────────────────────────────────────────────┐
*    │ _QSFOKP5UC           mbuttons VALID                      │
*    │                                                          │
*    │ Function Origin:                                         │
*    │                                                          │
*    │ From Platform:       MS-DOS                              │
*    │ From Screen:         CLAUSE,      Record Number:   24    │
*    │ Variable:            mbuttons                            │
*    │ Called By:           VALID Clause                        │
*    │ Object Type:         Push Button                         │
*    │ Snippet Number:      8                                   │
*    │                                                          │
*    └──────────────────────────────────────────────────────────┘
*
FUNCTION _qsf0kp5uc     &&  mbuttons VALID
#REGION 1
```

```
DO CASE
   *case mbuttons = 1   && delete
   *   m.sure = yesno("Sure you want to delete?","YES","NO")
   *   IF m.sure
   *      delete
   *      pack
   *   ENDIF
   *   m.id = 0
   *   m.op = " "
   *   m.val = " "
   *   mok = .f.
CASE mbuttons = 1   && ok
   mok = .T.
CASE mbuttons = 2   && cancel
   mok = .F.
ENDCASE


 *
 *
 *      CLAUSE/MS-DOS Supporting Procedures and Functions
 *
 *
 *

 #REGION 1
 FUNCTION insnewid
 PARAMETER mdata, mitem
 PRIVATE m.temp, m.search
 m.temp = ALLTRIM(mdata)
 m.search = .F.
 DO WHILE !EMPTY(m.temp)
    m.t1 = dp(m.temp, "|",1)
    m.temp = dp(m.temp,"|",2,999)
    IF VAL(m.t1) = mitem
       m.search = .T.
v──────EXIT
    ENDIF
 ENDDO
 IF !m.search
    mdata = ALLTRIM(mdata) + IIF(EMPTY(mdata),"","|") + ALLTRIM(STR(mi
=> tem))
 ENDIF
<══RETURN m.search

 FUNCTION setobject
 PARAMETER m.object, mtype
 IF "ACTION.DBF" $ DBF()
    m.object = "D"
    mtype = "Goal"
 ELSE
    m.object = "S"
    mtype = "Qualifier"
 ENDIF
<══RETURN
*:  EOF: CLAUSE.ac1
```

CLAUSE.AC1   10-3-94   3:01p

```
    *:**********************************************************************
=> *******
    *:
    *: Procedure file: C:\CAMD2\KBEDIT\WORKW\DP.PRG
    *:          System: Knowledge Base Editor
    *:          Author: Hoa L. Ly
    *:       Copyright (c) June 1, 1994, Naval Health Research Center, Code 2
=> 2
    *:  Last modified: 03/15/93 at  9:30:48
    *:
    *:          Set by: INSNEWID()         (function in CLAUSE.SPR)
    *:
    *:       Documented 15:01:02                                  FoxDoc versio
=> n 3.00a
    *:**********************************************************************
=> *******
      * ====================================================================
=> =================
      * Date: 08/04/92  10:01:15
      * Program Name: DP.PRG
      * Author's Name: Hoa Le Ly
      *
      * Copyright (c) 1992 Company Name: NHRC
      * Departament: Code 22
      * San Diego, CA 92138 - 5122
      * Description: This program emulate $PIECE of MUMPS function.  Which
=> return the
      *    the portion of string which is bounded by the characters in deli
=> miter. If both
      *    expr and expr2 are present, the value returned includes all char
=> acters from
      *     the expr1-1th occurrence of delimiter, up to but not including
=> the expr2th
      *    occurence of delimiter.  If expr2 is not present. it is assumed
=> to have the same
      *     value as expr.  If expr1 is not present. Then its value is assu
=> me to be 1
      * SYNTAX: DP(string, delimiter[,expr[,expr2]])
      * PARAMETER: string: Character expression which character extract fro
=> m
      *                  delimiter: Character to delimiter
      *                      expr: start of number occurrence of delimiter
      *                      expr2: number of occurrence delimiter
      * eg: string = "last, first^age^date of birth"
      *     ?dp(string, "^",1)           ===>  last, first
      *     ?dp(string,"^",1,2)          ===>  last, first^age
      * ====================================================================
=> ================

      PARAMETER ms,mm,mp,mp2
      PRIVATE ALL
    ┌─DO CASE
    ├─CASE PARAMETER() < 2
  <─────RETURN ""
    ├─CASE PARAMETER() = 2
    │    mp = 1
    │    mp2 = -1
    ├─CASE PARAMETER() = 3
    │  ┌─IF TYPE("mp") != "N"
  <──────────RETURN ""
    │  ├─ELSE
    │  │  ┌─IF mp < 1
  <──────────────RETURN ""
    │  │  └─ENDIF
    │  └─ENDIF
    │    mp2 = -1
    ├─CASE PARAMETER() = 4
    │  ┌─IF TYPE("mp") != "N"
  <──────────RETURN ""
    │  └─ENDIF
    │  ┌─IF TYPE("mp2") = "N"
    │  │    mp2 = IIF(mp >= mp2, -1, mp2)
    │  ├─ELSE
    │  │    mp2 = -1
    │  └─ENDIF
    └─ENDCASE
    ┌─IF TYPE("ms") != "C"
    │    ms = STR(ms)
    └─ENDIF
      moccurs = OCCURS(mm,ms)
    ┌─IF moccurs = 0                                         && Return ms if
=> mp=1&(ms'[mm)
    │    mstr = IIF(mp = 1, ms, "")                   && Return null if mp>1&(
=> ms'[mm)
  <─────RETURN mstr
    └─ENDIF
      mbegin = IIF(mp = 1, 1, (AT(mm,ms,(mp-1))+1))
    ┌─IF mp2 = -1
    │    mend = AT(mm,ms,mp) - 1
    │  ┌─IF mend < 0
    │  │  ┌─IF mbegin > 1
    │  │  │    mend = LEN(ms)
    │  │  ├─ELSE
  <────────────RETURN ""
    │  │  └─ENDIF
    │  └─ENDIF
    ├─ELSE
    │    mend = AT(mm,ms,mp2) - 1
    │    mend = IIF(mend<0,LEN(ms),mend)
    └─ENDIF
      mstr = SUBSTR(ms, mbegin, (mend - mbegin +1))
  <───RETURN mstr
   *:  EOF: DP.act
```

ICA4PRINT

```
*:*********************************************************************
=> *******
*:
*: Procedure file: C:\CAMD2\KBEDIT\WORKW\RESTORE.PRG
*:         System: Knowledge Base Editor
*:         Author: Hoa L. Ly
*:      Copyright (c) June 1, 1994, Naval Health Research Center, Code 2
=> 2
*:  Last modified: 06/03/94 at  9:26:30
*:
*:         Set by: KBMENU.MPR
*:
*:          Calls: RESTORE.SPR
*:
*:      Documented 15:01:02                              FoxDoc versio
=> n 3.00a
*:*********************************************************************
=> *******
    *-------------------------------------------------------------------
=> ----
    * Restore data files
    * Programmer: HLL
    * PROCEDURE restore
    *-------------------------------------------------------------------
=> ---
    PUBLIC mscreen
    PRIVATE MESSAGE
    SET TALK OFF
    SAVE SCREEN TO mscreen
    MESSAGE = "Restore Knowledge Base database"
    SET TOPIC TO "RESTORE"
    DO restore.spr WITH MESSAGE
    RESTORE SCREEN FROM mscreen
<===RETURN
*:  EOF: RESTORE.act
```

```
*
*       ┌────────────────────────────────────────────────────────┐
*       │ 08/09/94                RULE.SPR              09:39:29 │
*       ├────────────────────────────────────────────────────────┤
*       │                                                        │
*       │ Author's Name                                          │
*       │                                                        │
*       │ Copyright (c) 1994 Company Name                        │
*       │ Address                                                │
*       │ City,      Zip                                         │
*       │                                                        │
*       │ Description:                                           │
*       │ This program was automatically generated by GENSCRN.   │
*       │                                                        │
*       └────────────────────────────────────────────────────────┘
*

DO CASE
CASE _WINDOWS


   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
   ENDIF
   m.compstat = SET("COMPATIBLE")
   SET COMPATIBLE FOXPLUS

   m.rborder = SET("READBORDER")
   SET readborder ON

   *       ┌──────────────────────────────────────────────────┐
   *       │                                                  │
   *       │            Windows Window definitions            │
   *       │                                                  │
   *       └──────────────────────────────────────────────────┘
   *

   IF NOT WEXIST("w_rule") ;
         OR UPPER(WTITLE("W_RULE")) == "W_RULE.PJX" ;
         OR UPPER(WTITLE("W_RULE")) == "W_RULE.SCX" ;
         OR UPPER(WTITLE("W_RULE")) == "W_RULE.MNX" ;
         OR UPPER(WTITLE("W_RULE")) == "W_RULE.PRG" ;
         OR UPPER(WTITLE("W_RULE")) == "W_RULE.FRX" ;
         OR UPPER(WTITLE("W_RULE")) == "W_RULE.QPR"
      DEFINE WINDOW w_rule ;
         AT  0.000, 0.000  ;
         SIZE 25.333,64.375 ;
         TITLE "Rule Object Editor" ;
         FONT "Terminal", 8 ;
         FLOAT ;
         CLOSE ;
         SHADOW ;
         NOMINIMIZE ;
         COLOR RGB(,,,0,255,255)
      MOVE WINDOW w_rule CENTER
   ENDIF


   *       ┌──────────────────────────────────────────────────┐
   *       │                                                  │
   *       │     RULE/Windows Setup Code - SECTION 2          │
   *       │                                                  │
   *       └──────────────────────────────────────────────────┘
   *

   #REGION 1
   PRIVATE msel
   msel = SELECT()
   SELECT rule
   SET FILTER TO area = area.area
   SCATTER MEMO MEMVAR
   m.adding = .F.
   m.oldrec = RECNO()

   *       ┌──────────────────────────────────────────────────┐
   *       │                                                  │
   *       │          RULE/Windows Screen Layout              │
   *       │                                                  │
   *       └──────────────────────────────────────────────────┘
   *

   #REGION 1
   IF WVISIBLE("w_rule")
      ACTIVATE WINDOW w_rule SAME
   ELSE
      ACTIVATE WINDOW w_rule NOSHOW
   ENDIF
   @ 1.250,1.500 SAY "Rule"  ;
      FONT "Terminal", 8
   @ 1.250,6.625 GET m.rule ;
      SIZE 1.000,6.000 ;
      DEFAULT 0 ;
      FONT "Terminal", 8 ;
      DISABLE
   @ 1.250,15.500 SAY "Salience"  ;
      FONT "Terminal", 8
   @ 1.250,25.125 GET m.salience ;
      SIZE 1.000,5.000 ;
      DEFAULT " " ;
      FONT "Terminal", 8 ;
      VALID m.salience >= 0 .AND. m.salience <=  10000
   @ 22.667,24.000 GET mbuttons ;
      PICTURE "@*HT \<Ok ;\<Cancel" ;
      SIZE 1.917,7.500,0.750 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kp97y()
   @ 3.333,1.500 SAY "Explanation"  ;
      FONT "Terminal", 8
   @ 13.250,1.250 SAY "Note"  ;
      FONT "Terminal", 8
      @ 4.750,1.500 EDIT m.explain ;
         SIZE 7.583,60.375,0.000 ;
         DEFAULT " " ;
         FONT "Terminal", 8 ;
         SCROLL
      @ 14.500,1.500 EDIT m.note ;
         SIZE 7.333,60.125,0.000 ;
         DEFAULT " " ;
         FONT "Terminal", 8 ;
         SCROLL

   ┌IF NOT WVISIBLE("w_rule")
   │   ACTIVATE WINDOW w_rule
   └ENDIF

      READ CYCLE MODAL

      RELEASE WINDOW w_rule

      #REGION 0

      SET readborder &rborder

   ┌IF m.talkstat = "ON"
   │   SET TALK ON
   └ENDIF
   ┌IF m.compstat = "ON"
   │   SET COMPATIBLE ON
   └ENDIF


      *      ┌─────────────────────────────────────
      *      │
      *      │        RULE/Windows Cleanup Code
      *      │
      *      └─────────────────────────────────────

      *


      #REGION 1
      SELECT rule
      SET FILTER TO

 ─CASE _DOS


      #REGION 0
      REGIONAL m.currarea, m.talkstat, m.compstat

   ┌IF SET("TALK") = "ON"
   │   SET TALK OFF
   │   m.talkstat = "ON"
   ├ELSE
   │   m.talkstat = "OFF"
   └ENDIF
      m.compstat = SET("COMPATIBLE")
      SET COMPATIBLE FOXPLUS

      *      ┌─────────────────────────────────────
      *      │
      *      │        MS-DOS Window definitions
      *      │
      *      └─────────────────────────────────────

      *

   ┌IF NOT WEXIST("w_rule") ;
   │      OR UPPER(WTITLE("W_RULE")) == "W_RULE.PJX" ;
   │      OR UPPER(WTITLE("W_RULE")) == "W_RULE.SCX" ;
   │      OR UPPER(WTITLE("W_RULE")) == "W_RULE.MNX" ;
   │      OR UPPER(WTITLE("W_RULE")) == "W_RULE.PRG" ;
   │      OR UPPER(WTITLE("W_RULE")) == "W_RULE.FRX" ;
   │      OR UPPER(WTITLE("W_RULE")) == "W_RULE.QPR"
   │   DEFINE WINDOW w_rule ;
   │      FROM INT((SROW()-20)/2),INT((SCOL()-76)/2) ;
   │      TO INT((SROW()-20)/2)+19,INT((SCOL()-76)/2)+75 ;
   │      TITLE "Rule Object Editor" ;
   │      FLOAT ;
   │      CLOSE ;
   │      SHADOW ;
   │      NOMINIMIZE ;
   │      COLOR SCHEME 1
   └ENDIF


      *      ┌─────────────────────────────────────
      *      │
      *      │   RULE/MS-DOS Setup Code - SECTION 2
      *      │
      *      └─────────────────────────────────────

      *


      #REGION 1
      PRIVATE msel
      msel = SELECT()
      SELECT rule
      SET FILTER TO area = area.area
      SCATTER MEMO MEMVAR
      m.adding = .F.
      m.oldrec = RECNO()

      *      ┌─────────────────────────────────────
      *      │
      *      │        RULE/MS-DOS Screen Layout
      *      │
      *      └─────────────────────────────────────

      *


      #REGION 1
   ┌IF WVISIBLE("w_rule")
   │   ACTIVATE WINDOW w_rule SAME
   ├ELSE
   │   ACTIVATE WINDOW w_rule NOSHOW
   └ENDIF
   @ 1,1 SAY "Rule" ;
      SIZE 1,4, 0
   @ 1,6 GET m.rule ;
      SIZE 1,5 ;
      DEFAULT 0 ;
      DISABLE
   @ 1,15 SAY "Salience" ;
      SIZE 1,8, 0
   @ 1,24 GET m.salience ;
      SIZE 1,4 ;
      DEFAULT " " ;
      VALID m.salience >= 0 .AND. m.salience <=  10000
   @ 17,15 GET mbbutton ;
      PICTURE "@*VN \<Add" ;
      SIZE 1,8,1 ;
      DEFAULT 1 ;
      VALID _qsf0kpa3b()
   @ 17,25 GET mbuttons ;
      PICTURE "@*HT \<Delete ;\<Ok ;\<Cancel" ;
      SIZE 1,9,1 ;
      DEFAULT 1 ;
      VALID _qsf0kpa7j()
   @ 3,1 SAY "Explanation" ;
      SIZE 1,11, 0
   @ 10,1 SAY "Note" ;
      SIZE 1,4, 0
   @ 4,1 EDIT m.explain ;
      SIZE 6,72,0 ;
      DEFAULT " " ;
      SCROLL
   @ 11,1 EDIT m.note ;
      SIZE 6,72,0 ;
      DEFAULT " " ;
      SCROLL

  IF NOT WVISIBLE("w_rule")
      ACTIVATE WINDOW w_rule
  ENDIF

   READ CYCLE MODAL

   RELEASE WINDOW w_rule

   #REGION 0
  IF m.talkstat = "ON"
      SET TALK ON
  ENDIF
  IF m.compstat = "ON"
      SET COMPATIBLE ON
  ENDIF


   *
   *
   *                  RULE/MS-DOS Cleanup Code
   *
   *
   *

   #REGION 1
   SELECT rule
      SET FILTER TO

  ENDCASE


*
*
*        _QSFOKP97Y           mbuttons VALID
*
*        Function Origin:
*
*        From Platform:       Windows
*        From Screen:         RULE,      Record Number:    6
*        Variable:            mbuttons
*        Called By:           VALID Clause
*        Object Type:         Push Button
*        Snippet Number:      1
*
*
*
FUNCTION _qsf0kp97y     &&  mbuttons VALID
#REGION 1
DO CASE
CASE mbuttons = 1   && ok
   GATHER MEMVAR MEMO
CASE mbuttons = 2   && cancel
   SCATTER MEMO MEMVAR
ENDCASE

*
*
*        _QSFOKPA3B           mbbutton VALID
*
*        Function Origin:
*
*        From Platform:       MS-DOS
*        From Screen:         RULE,      Record Number:    17
*        Variable:            mbbutton
*        Called By:           VALID Clause
*        Object Type:         Push Button
*        Snippet Number:      2
*
*
*
FUNCTION _qsf0kpa3b     &&  mbbutton VALID
#REGION 1
m.adding = .T.
m.oldrec = RECNO()
SET TOPIC TO "ADD"
SCATTER MEMO MEMVAR BLANK
m.area = area.area
GOTO BOTTOM
m.rule = rule + 1
m.order = SET("order")
SET ORDER TO salience
GOTO BOTTOM
m.salience = salience + 10
SET ORDER TO (m.order)
SET TOPIC TO "RULE"
SHOW GET mbbutton disabled
SHOW GETS


*
*
*        _QSFOKPA7J           mbuttons VALID
```

```
*
*           Function Origin:
*
*           From Platform:       MS-DOS
*           From Screen:         RULE,      Record Number:   18
*           Variable:            mbuttons
*           Called By:           VALID Clause
*           Object Type:         Push Button
*           Snippet Number:      3
*
*
*
FUNCTION _qsf0kpa7j     &&  mbuttons VALID
#REGION 1
DO CASE
CASE mbuttons = 1   && delete
   m.del = yesno("Sure You want delete?","YES","NO")
   IF m.del
      m.sel = SELECT()
      SET TOPIC TO "DELETE"
      SELECT premise
      DELETE FOR clause = m.premise
      PACK
      SELECT action
      DELETE FOR clause = m.action
      PACK
      SELECT rule
      DELETE
      PACK
      COUNT TO m.rules
      SELECT area
      REPLACE rules WITH m.rules
      SELECT (m.sel)
   ENDIF
   SCATTER MEMO MEMVAR
   SET TOPIC TO "RULE"
CASE mbuttons = 2   && ok
   IF m.adding
      APPEND BLANK
      GATHER MEMVAR MEMO
      m.recno = RECNO()
      COUNT TO m.rules
      GOTO m.recno
      m.sel = SELECT()
      SELECT area
      REPLACE rules WITH m.rules
      SELECT (m.sel)
   ELSE
      GATHER MEMVAR MEMO
   ENDIF
CASE mbuttons = 3   && cancel
   GOTO m.oldrec
   SCATTER MEMO MEMVAR
ENDCASE
*:  EOF: RULE.ac1
```

P-4

```
*
*
*    08/09/94              TERM.SPR              09:39:33
*
*
*    Author's Name
*
*    Copyright (c) 1994 Company Name
*    Address
*    City,     Zip
*
*    Description:
*    This program was automatically generated by GENSCRN.
*
*

DO CASE
CASE _WINDOWS


   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
   ENDIF
   m.compstat = SET("COMPATIBLE")
   SET COMPATIBLE FOXPLUS

   m.rborder = SET("READBORDER")
   SET readborder ON

   *
   *
   *         Windows Window definitions
   *
   *
   *

   IF NOT WEXIST("w_term") ;
         OR UPPER(WTITLE("W_TERM")) == "W_TERM.PJX" ;
         OR UPPER(WTITLE("W_TERM")) == "W_TERM.SCX" ;
         OR UPPER(WTITLE("W_TERM")) == "W_TERM.MNX" ;
         OR UPPER(WTITLE("W_TERM")) == "W_TERM.PRG" ;
         OR UPPER(WTITLE("W_TERM")) == "W_TERM.FRX" ;
         OR UPPER(WTITLE("W_TERM")) == "W_TERM.QPR"
      DEFINE WINDOW w_term ;
         AT  0.000, 0.000  ;
         SIZE 19.917,55.250 ;
         TITLE "Term [IF] Editor" ;
         FONT "Terminal", 8 ;
         FLOAT ;
         CLOSE ;
         SHADOW ;
         NOMINIMIZE ;
         COLOR RGB(,,,0,255,255)
      MOVE WINDOW w_term CENTER
   ENDIF


   *
   *
   *         TERM/Windows Setup Code - SECTION 2
   *
   *
   *

   #REGION 1
   EXTERNAL ARRAY prem, premr, premo, triple
   PRIVATE mselect, mok
   mselect = SELECT()
   SELECT premise
   IF EOF()
      PRIVATE m.flag = .F.
      GOTO BOTTOM
      m.clause = IIF(EOF(),0, clause) + 1
      m.flag = addfact()
      IF m.flag
         PRIVATE m.sel
         m.sel = SELECT()
         SELECT rule
         REPLACE premise WITH m.clause
         SELECT (m.sel)
      ENDIF
   ENDIF


   *
   *
   *         TERM/Windows Screen Layout
   *
   *
   *

   #REGION 1
   IF WVISIBLE("w_term")
      ACTIVATE WINDOW w_term SAME
   ELSE
      ACTIVATE WINDOW w_term NOSHOW
   ENDIF
   @ 1.250,3.375 GET mp ;
      PICTURE "@&N" ;
      FROM prem ;
      SIZE 16.333,36.875 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kpc3r()
   @ 2.833,43.125 GET mbuttons ;
      PICTURE "@*VN \<Edit;\<Quit" ;
      SIZE 4.417,8.000,1.083 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kpc6d()
```

```
    IF NOT WVISIBLE("w_term")
        ACTIVATE WINDOW w_term
    ENDIF

    READ CYCLE MODAL

    RELEASE WINDOW w_term

    #REGION 0

    SET readborder &rborder

    IF m.talkstat = "ON"
        SET TALK ON
    ENDIF
    IF m.compstat = "ON"
        SET COMPATIBLE ON
    ENDIF


     *
     *
     *          TERM/Windows Cleanup Code
     *
     *
     *

     #REGION 1
<====RETURN .T.


  CASE _DOS


     #REGION 0
     REGIONAL m.currarea, m.talkstat, m.compstat

    IF SET("TALK") = "ON"
        SET TALK OFF
        m.talkstat = "ON"
    ELSE
        m.talkstat = "OFF"
    ENDIF
    m.compstat = SET("COMPATIBLE")
    SET COMPATIBLE FOXPLUS

     *
     *
     *          MS-DOS Window definitions
     *
     *
     *

    IF NOT WEXIST("w_term") ;
            OR UPPER(WTITLE("W_TERM")) == "W_TERM.PJX" ;
            OR UPPER(WTITLE("W_TERM")) == "W_TERM.SCX" ;
            OR UPPER(WTITLE("W_TERM")) == "W_TERM.MNX" ;
            OR UPPER(WTITLE("W_TERM")) == "W_TERM.PRG" ;
            OR UPPER(WTITLE("W_TERM")) == "W_TERM.FRX" ;
            OR UPPER(WTITLE("W_TERM")) == "W_TERM.QPR"
        DEFINE WINDOW w_term ;
            FROM INT((SROW()-19)/2),INT((SCOL()-76)/2) ;
            TO INT((SROW()-19)/2)+18,INT((SCOL()-76)/2)+75 ;
            TITLE "Term Editor" ;
            FLOAT ;
            CLOSE ;
            SHADOW ;
            NOMINIMIZE ;
            COLOR SCHEME 1
    ENDIF


     *
     *
     *          TERM/MS-DOS Setup Code - SECTION 2
     *
     *
     *

     #REGION 1
     EXTERNAL ARRAY prem, premr, premo, triple
     PRIVATE mselect, mok
     mselect = SELECT()
     SELECT premise
    IF EOF()
        PRIVATE m.flag = .F.
        GOTO BOTTOM
        m.clause = IIF(EOF(),0, clause) + 1
        m.flag = addfact()
        IF m.flag
            PRIVATE m.sel
            m.sel = SELECT()
            SELECT rule
            REPLACE premise WITH m.clause
            SELECT (m.sel)
        ENDIF
    ENDIF


     *
     *
     *          TERM/MS-DOS Screen Layout
     *
     *
     *

     #REGION 1
    IF WVISIBLE("w_term")
        ACTIVATE WINDOW w_term SAME
    ELSE
        ACTIVATE WINDOW w_term NOSHOW
  └ENDIF
   @ 9,63 GET mbuttons ;
      PICTURE "@*VT \<Delete;\<OK;\<Cancel" ;
      SIZE 1,8,1 ;
      DEFAULT 1 ;
      VALID _qsf0kpcpm()
   @ 3,63 GET mbbutton ;
      PICTURE "@*VN \<AND" ;
      SIZE 1,8,1 ;
      DEFAULT 1 ;
      VALID _qsf0kpcu9()
   @ 5,63 GET morbutton ;
      PICTURE "@*VN \<OR" ;
      SIZE 1,8,1 ;
      DEFAULT 1 ;
      VALID _qsf0kpcwu()
   @ 1,63 GET mebutton ;
      PICTURE "@*VN \<Edit" ;
      SIZE 1,8,1 ;
      DEFAULT 1 ;
      WHEN mp > 0 ;
      VALID _qsf0kpczg()
   @ 7,63 GET minsbutton ;
      PICTURE "@*VN \<Insert" ;
      SIZE 1,8,1 ;
      DEFAULT 1 ;
      VALID _qsf0kpd1z()
   @ 0,1 GET mp ;
      PICTURE "@&N" ;
      FROM prem ;
      SIZE 17,59 ;
      DEFAULT 1 ;
      VALID _qsf0kpd51() ;
      COLOR SCHEME 2

  ┌IF NOT WVISIBLE("w_term")
  │   ACTIVATE WINDOW w_term
  └ENDIF

   READ CYCLE MODAL

   RELEASE WINDOW w_term

   #REGION 0
  ┌IF m.talkstat = "ON"
  │   SET TALK ON
  └ENDIF
  ┌IF m.compstat = "ON"
  │   SET COMPATIBLE ON
  └ENDIF


   *
   *
   *            TERM/MS-DOS Cleanup Code
   *
   *
   *

   #REGION 1
<──RETURN .T.
```

`TERM.AC1   10-3-94   3:01p`

```
 │
 │
 └ENDCASE


   *
   *
   *     _QSFOKPC3R            mp VALID
   *
   *     Function Origin:
   *
   *     From Platform:        Windows
   *     From Screen:          TERM,     Record Number:    2
   *     Variable:             mp
   *     Called By:            VALID Clause
   *     Object Type:          List
   *     Snippet Number:       1
   *
   *
   *
   FUNCTION _qsf0kpc3r     &&  mp VALID
   #REGION 1
   DO edprem

   *
   *
   *     _QSFOKPC6D            mbuttons VALID
   *
   *     Function Origin:
   *
   *     From Platform:        Windows
   *     From Screen:          TERM,     Record Number:    3
   *     Variable:             mbuttons
   *     Called By:            VALID Clause
   *     Object Type:          Push Button
   *     Snippet Number:       2
   *
   *
   *
   FUNCTION _qsf0kpc6d     &&  mbuttons VALID
   #REGION 1
  ┌DO CASE
  ├CASE mbuttons = 1   && Edit
  │  ┌IF mp > 0
  │  │   DO edprem
  │  └ENDIF
  │   mok = .T.
  ├CASE mbuttons = 2   && cancel
  │   mok = .F.
  │   CLEAR READ
  └ENDCASE

   *
   *
   *     _QSFOKPCPM            mbuttons VALID
   *
   *     Function Origin:
   *
   *     From Platform:        MS-DOS
   *     From Screen:          TERM,     Record Number:    6
   *     Variable:             mbuttons
   *     Called By:            VALID Clause
   *     Object Type:          Push Button
   *     Snippet Number:       3
   *
   *
```

```
*
FUNCTION _qsf0kpcpm     &&  mbuttons VALID
#REGION 1
DO CASE
CASE mbuttons = 1         && DELETE
   SEEK rule.premise
   DO WHILE clause = rule.premise
      IF EMPTY(premise.op)
         SELECT fact
         DELETE FOR clause = premise.fact
         SELECT premise
      ENDIF
      DELETE
      SKIP
   ENDDO
   SELECT fact
   *  pack                        && wait till the end
   SELECT premise
   *  pack                        && wait till the end
   mok = .F.
CASE mbuttons = 2   && ok
   mok = .T.
CASE mbuttons = 3   && cancel
   mok = .F.
ENDCASE

*
*
*     _QSFOKPCU9           mbbutton VALID
*
*     Function Origin:
*
*     From Platform:       MS-DOS
*     From Screen:         TERM,      Record Number:    7
*     Variable:            mbbutton
*     Called By:           VALID Clause
*     Object Type:         Push Button
*     Snippet Number:      4
*
*
*
FUNCTION _qsf0kpcu9     &&  mbbutton VALID
#REGION 1
=setjoin("&")

*
*
*     _QSFOKPCWU           morbutton VALID
*
*     Function Origin:
*
*     From Platform:       MS-DOS
*     From Screen:         TERM,      Record Number:    8
*     Variable:            morbutton
*     Called By:           VALID Clause
*     Object Type:         Push Button
*     Snippet Number:      5
*
*
*
FUNCTION _qsf0kpcwu     &&  morbutton VALID
#REGION 1
=setjoin("|")

*
*
*     _QSFOKPCZG           mebutton VALID
*
*     Function Origin:
*
*     From Platform:       MS-DOS
*     From Screen:         TERM,      Record Number:    9
*     Variable:            mebutton
*     Called By:           VALID Clause
*     Object Type:         Push Button
*     Snippet Number:      6
*
*
*
FUNCTION _qsf0kpczg     &&  mebutton VALID
#REGION 1
DO edprem

*
*
*     _QSFOKPD1Z           minsbutton VALID
*
*     Function Origin:
*
*     From Platform:       MS-DOS
*     From Screen:         TERM,      Record Number:    10
*     Variable:            minsbutton
*     Called By:           VALID Clause
*     Object Type:         Push Button
*     Snippet Number:      7
*
*
*
FUNCTION _qsf0kpd1z     &&  minsbutton VALID
#REGION 1
m.clause = rule.premise
= addfact()
SHOW GET mbbutton ENABLE
SHOW GETS



*
*
*     _QSFOKPD51           mp VALID
*
*     Function Origin:
*
*     From Platform:       MS-DOS
*     From Screen:         TERM,      Record Number:    11
*     Variable:            mp
*     Called By:           VALID Clause
*     Object Type:         List
*     Snippet Number:      8
*
*
*
FUNCTION _qsf0kpd51     &&  mp VALID
#REGION 1
DO edprem


*
*
*     TERM/MS-DOS Supporting Procedures and Functions
*
*
```

Q-4

IC/14PRINT

```
  *

  #REGION 1
  PROCEDURE edprem
  EXTERNAL ARRAY prem, premr, triple
  SET TOPIC TO "EDIT"
  SELECT fact
  K = premr[mp]
┌─IF triple[k,2] > 0
│    SEEK triple[k,2]
├─ELSE
│    * empty fact list; add insert fact !!!
└─ENDIF
  m.clause = premise.fact
  DO clause.spr
  SELECT premise
  DO setprem
  SHOW GETS
  mtopic = ALIAS()
  SET TOPIC TO &mtopic
<──RETURN


  FUNCTION setjoin
  PARAMETERS joinop
┌─IF mp > 0
│    K = VAL(premo[mp])
│  ┌─IF K > 0
│  │    triple[k,1] = joinop
│  │  ┌─IF triple[k,5] > 0
│  │  │    GOTO triple[k,5]
│  │  │    REPLACE premise.op WITH joinop
│  │  └─ENDIF
│  └─ENDIF
└─ENDIF
  DO setprem
  SHOW GETS
<──RETURN joinop

  FUNCTION lastfact
  SELECT premise
┌─IF EOF()
│    m.fact = 0
├─ELSE
│    m.fact = fact
└─ENDIF
  SELECT fact
<──RETURN m.fact

  FUNCTION addfact
  PRIVATE m.sel, m.flag
  m.sel = SELECT()
  m.flag = .F.
  SELECT fact
  GOTO BOTTOM
  m.fact = IIF(EOF(), 0, clause) + 1
  DO clause.spr WITH m.fact
  SEEK m.fact
┌─IF FOUND()
│    SELECT premise
│    APPEND BLANK
│    REPLACE clause WITH m.clause
│    REPLACE fact WITH m.fact
│    m.flag = .T.
└─ENDIF
  SELECT (m.sel)
  DO setprem
<──RETURN m.flag
 *:  EOF: TERM.ac1
```

R-1

```
    *:*********************************************************************
=> *******
    *:
    *: Procedure file: C:\CAMD2\KBEDIT\WORKW\KBDELETE.PRG
    *:         System: Knowledge Base Editor
    *:         Author: Hoa L. Ly
    *:      Copyright (c) June 1, 1994, Naval Health Research Center, Code 2
=> 2
    *:  Last modified: 07/27/94 at 10:47:58
    *:
    *:         Set by: _QSFOKOKEU()       (function in KBDEL.SPR)
    *:               : _QSFOKOKPT()       (function in KBDEL.SPR)
    *:
    *:           Uses: FACT.DBF
    *:               : PREMISE.DBF
    *:               : ACTION.DBF
    *:               : RULE.DBF
    *:
    *:        Indexes: FACT.IDX
    *:               : PREMISE.IDX
    *:               : ACTION.IDX
    *:               : RULEAREA.IDX
    *:               : SALIENCE.IDX
    *:               : RULE.IDX
    *:
    *:     Documented 15:01:04                                  FoxDoc versio
=> n 3.00a
    *:*********************************************************************
=> *******
        * kbdelete.prg

        PARAMETERS m.name                       && name of knowledge base

        * Removes rules and their associated premise, action clauses
        * from the knowledge base.

        * Lookup the name in the knowledge base file
        SELECT area                             &&
        LOCATE FOR LOWER(name) = LOWER(m.name)  && lookup by name (case inse
=> nsitive)
      ┌IF !FOUND()                              &&
      │   * unknown knowledge base              &&
  <───RETURN                                    &&
      └ENDIF                                    &&

        * open database
        SELECT 0
        USE fact INDEX fact
        SELECT 0
        USE premise INDEX premise
        SET RELATION TO fact INTO fact
        SELECT 0
        USE action INDEX action
        SELECT 0
        USE rule INDEX rulearea,salience,rule
        SET RELATION TO premise INTO premise, action INTO action
        SELECT area
        SET RELATION TO area INTO rule

        * delete rules and clauses in knowledge base
        SELECT rule                              &&
        SEEK area.area
      ┌DO WHILE area = area.area .AND. !EOF()
      │  SELECT premise
      │  SEEK rule.premise
      │ ┌DO WHILE clause = rule.premise
      │ │  ┌IF EMPTY(premise.op)
      │ │  │   SELECT fact
      │ │  │   DELETE FOR clause = premise.fact   && delete all premise cla
=> uses
      │ │  └ENDIF
      │ │    SELECT premise
      │ │    DELETE
      │ │    SKIP
      │ └ENDDO
      │   SELECT action
      │   DELETE FOR clause = rule.action      && delete all action clauses
      │   DELETE FOR clause = rule.else        && delete all else clauses
      │   SELECT rule
      │   DELETE                               && delete the rule
      │   SKIP
      └ENDDO

        * delete goals and qualifiers in knowledge base
        SELECT goals
        DELETE FOR area = area.area
        PACK
        SELECT DISPLAY
        DELETE FOR area = area.area
        PACK
        SELECT quals
        DELETE FOR area = area.area
        PACK

        * update premise database
        SELECT premise
        PACK                                   && purge premises

        * update fact database
        SELECT fact
        PACK                                   && purge facts

        * update action database
        SELECT action
        PACK                                   && purge actions

        * update rulebase
        SELECT rule                            &&
        PACK                                   && purge rules

        * update area database
        SELECT area
        *replace rules with 0                   && indicate no rules
        PRIVATE m.recno
        m.recno = RECNO()
        DELETE
        PACK
      ┌IF m.recno > RECCOUNT()
      │   GOTO BOTTOM
      ├ELSE
      │   GOTO m.recno
      └ENDIF

        *close unuse database
        SELECT fact
        USE
        SELECT premise
        USE
        SELECT action
        USE
        SELECT rule
        USE

    * finish up
<——RETURN

 *:  EOF: KBDELETE.act
```

```
*
*       08/09/94            OBJECT.SPR                09:39:38
*
*
*
*       Author's Name
*
*       Copyright (c) 1994 Company Name
*       Address
*       City,     Zip
*
*       Description:
*       This program was automatically generated by GENSCRN.
*
*

PARAMETERS m.status
DO CASE
CASE _WINDOWS


   *
   *
   *            OBJECT/Windows Setup Code - SECTION 1
   *
   *
   *

   #REGION 1
   IF PARAMETER() = 0
      m.status = "Q"
   ENDIF
   m.sel = SELECT()
   SELECT dict
   GOTO BOTTOM
   m.id = id + 1
   m.get = .F.
   DIMENSION marray[1]

   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
   ENDIF
   m.compstat = SET("COMPATIBLE")
   SET COMPATIBLE FOXPLUS

   m.rborder = SET("READBORDER")
   SET readborder ON

   *
   *
   *                  Windows Window definitions
   *
   *
   *

   IF NOT WEXIST("object") ;
         OR UPPER(WTITLE("OBJECT")) == "OBJECT.PJX" ;
         OR UPPER(WTITLE("OBJECT")) == "OBJECT.SCX" ;
         OR UPPER(WTITLE("OBJECT")) == "OBJECT.MNX" ;
         OR UPPER(WTITLE("OBJECT")) == "OBJECT.PRG" ;
         OR UPPER(WTITLE("OBJECT")) == "OBJECT.FRX" ;
         OR UPPER(WTITLE("OBJECT")) == "OBJECT.QPR"
      DEFINE WINDOW OBJECT ;
         AT  0.000, 0.000  ;
         SIZE 12.769,54.333 ;
         TITLE "Add New Object" ;
         FONT "MS Sans Serif", 8 ;
         STYLE "B" ;
         NOFLOAT ;
         NOCLOSE ;
         SHADOW ;
         NOMINIMIZE ;
         COLOR RGB(,,,128,128,0)
      MOVE WINDOW OBJECT CENTER
   ENDIF


   *
   *
   *                  OBJECT/Windows Screen Layout
   *
   *
   *

   #REGION 1
   IF WVISIBLE("object")
      ACTIVATE WINDOW OBJECT SAME
   ELSE
      ACTIVATE WINDOW OBJECT NOSHOW
   ENDIF
   @ 7.385,3.000 SAY "ID# :"  ;
      FONT "MS Sans Serif", 8 ;
      STYLE "B"
   @ 1.462,10.333 GET m.object ;
      PICTURE "@*RHN Subject     ;Disease" ;
      SIZE 1.308,15.000,0.000 ;
      DEFAULT 1 ;
      FONT "MS Sans Serif", 8 ;
      STYLE "B" ;
      VALID  qsf0kpfr1()
   @ 4.385,12.333 GET m.name ;
      SIZE 1.154,38.000 ;
      DEFAULT " " ;
      FONT "MS Sans Serif", 8 ;
      VALID  qsf0kpfv8()
   @ 7.385,10.500 GET m.id ;
      SIZE 1.154,5.000 ;
      DEFAULT 0 ;
      FONT "MS Sans Serif", 8 ;
      DISABLE
```

S-1

```
      @ 9.692,13.167 GET m.button ;
         PICTURE "@*HT \<Ok;\<Cancel" ;
         SIZE 1.692,9.833,1.000 ;
         DEFAULT 1 ;
         FONT "MS Sans Serif", 8 ;
         STYLE "B" ;
         VALID _qsf0kpfyy()
      @ 1.615,3.000 SAY "Type:" ;
         FONT "MS Sans Serif", 8 ;
         STYLE "B"
      @ 4.231,3.000 GET m.obj ;
         PICTURE "@*HN Object" ;
         SIZE 1.769,8.000,1.000 ;
         DEFAULT 1 ;
         FONT "MS Sans Serif", 8 ;
         STYLE "B" ;
         VALID _qsf0kpg47()

      IF NOT WVISIBLE("object")
         ACTIVATE WINDOW OBJECT
      ENDIF

      READ CYCLE MODAL

      RELEASE WINDOW OBJECT

      #REGION 0

      SET readborder &rborder

      IF m.talkstat = "ON"
         SET TALK ON
      ENDIF
      IF m.compstat = "ON"
         SET COMPATIBLE ON
      ENDIF

   CASE _DOS

      *
      *
      *         OBJECT/MS-DOS Setup Code - SECTION 1
      *
      *
      *

      #REGION 1
      IF PARAMETER() = 0
         m.status = "Q"
      ENDIF
      m.sel = SELECT()
      SELECT dict
      GOTO BOTTOM
      m.id = id + 1
      m.get = .F.
      DIMENSION marray[1]

      #REGION 0
      REGIONAL m.currarea, m.talkstat, m.compstat

      IF SET("TALK") = "ON"
         SET TALK OFF
         m.talkstat = "ON"
      ELSE
         m.talkstat = "OFF"
      ENDIF
      m.compstat = SET("COMPATIBLE")
      SET COMPATIBLE FOXPLUS

      *
      *
      *         MS-DOS Window definitions
      *
      *
      *

      IF NOT WEXIST("object") ;
            OR UPPER(WTITLE("OBJECT")) == "OBJECT.PJX" ;
            OR UPPER(WTITLE("OBJECT")) == "OBJECT.SCX" ;
            OR UPPER(WTITLE("OBJECT")) == "OBJECT.MNX" ;
            OR UPPER(WTITLE("OBJECT")) == "OBJECT.PRG" ;
            OR UPPER(WTITLE("OBJECT")) == "OBJECT.FRX" ;
            OR UPPER(WTITLE("OBJECT")) == "OBJECT.QPR"
         DEFINE WINDOW OBJECT ;
            FROM INT((SROW()-11)/2),INT((SCOL()-54)/2) ;
            TO INT((SROW()-11)/2)+10,INT((SCOL()-54)/2)+53 ;
            TITLE "Add New Object" ;
            NOFLOAT ;
            NOCLOSE ;
            SHADOW ;
            NOMINIMIZE ;
            COLOR SCHEME 1
      ENDIF

      *
      *
      *         OBJECT/MS-DOS Screen Layout
      *
      *
      *

      #REGION 1
      IF WVISIBLE("object")
         ACTIVATE WINDOW OBJECT SAME
      ELSE
         ACTIVATE WINDOW OBJECT NOSHOW
      ENDIF
      @ 6,1 SAY "ID# :" ;
         SIZE 1,5, 0
      @ 2,8 GET m.object ;
         PICTURE "@*RHN Subject     ;Disease" ;
         SIZE 1,16,0 ;
         DEFAULT 1 ;
         VALID _qsf0kpgoj()
```

```
│   @ 4,10 GET m.name ;
│      SIZE 1,38 ;
│      DEFAULT " " ;
│      VALID _qsf0kpgsr()
│   @ 6,8 GET m.id ;
│      SIZE 1,5 ;
│      DEFAULT 0 ;
│      DISABLE
│   @ 8,19 GET m.button ;
│      PICTURE "@*HT \<Ok;\<Cancel" ;
│      SIZE 1,8,1 ;
│      DEFAULT 1 ;
│      VALID _qsf0kpgvz()
│   @ 2,1 SAY "Type:" ;
│      SIZE 1,5, 0
│   @ 4,1 GET m.obj ;
│      PICTURE "@*HN Object" ;
│      SIZE 1,8,1 ;
│      DEFAULT 1 ;
│      VALID _qsf0kph17()
│
│ ┌IF NOT WVISIBLE("object")
│ │  ACTIVATE WINDOW OBJECT
│ └ENDIF
│
│   READ CYCLE MODAL
│
│   RELEASE WINDOW OBJECT
│
│   #REGION 0
│ ┌IF m.talkstat = "ON"
│ │  SET TALK ON
│ └ENDIF
│ ┌IF m.compstat = "ON"
│ │  SET COMPATIBLE ON
│ └ENDIF
│
│
└ENDCASE


  *
  *
  *       _QSFOKPFR1          m.object VALID
  *
  *       Function Origin:
  *
  *       From Platform:      Windows
  *       From Screen:        OBJECT,     Record Number:    3
  *       Variable:           m.object
  *       Called By:          VALID Clause
  *       Object Type:        Radio Button
  *       Snippet Number:     1
  *
  *
  *
  FUNCTION _qsf0kpfr1     &&  m.object VALID
  #REGION 1
┌IF m.object = 1
│   SELECT dict
│   SET ORDER TO 1
│   GOTO BOTTOM
│   m.id = id + 1
├ELSE
│ ┌IF m.object = 2
│ │   SELECT disease
│ │   SET ORDER TO 1
│ │   GOTO BOTTOM
│ │   m.id = id + 1
│ └ENDIF
└ENDIF
  SHOW GETS

  *
  *
  *       _QSFOKPFV8          m.name VALID
  *
  *       Function Origin:
  *
  *       From Platform:      Windows
  *       From Screen:        OBJECT,     Record Number:    4
  *       Variable:           m.name
  *       Called By:          VALID Clause
  *       Object Type:        Field
  *       Snippet Number:     2
  *
  *
  *
  FUNCTION _qsf0kpfv8     &&  m.name VALID
  #REGION 1
  =getobj(m.name)

  *
  *
  *       _QSFOKPFYY          m.button VALID
  *
  *       Function Origin:
  *
  *       From Platform:      Windows
  *       From Screen:        OBJECT,     Record Number:    6
  *       Variable:           m.button
  *       Called By:          VALID Clause
  *       Object Type:        Push Button
  *       Snippet Number:     3
  *
  *
  *
  FUNCTION _qsf0kpfyy     &&  m.button VALID
  #REGION 1
┌IF m.button = 1
│   * SEEK m.id
│   * IF !found()
│   APPEND BLANK
│   REPLACE id WITH m.id
│   REPLACE name WITH m.name
│   *       IF !m.get
│   *          = datainput()
│   *       ENDIF
│   * ENDIF
│ ┌DO CASE
│ ├CASE m.status = "Q"
│ │    SELECT quals
│ ├CASE m.status = "G"
│ │    SELECT goals
│ └ENDCASE
│   APPEND BLANK
│   REPLACE id WITH m.id
│   REPLACE OBJECT WITH IIF(m.object=1,"S","D")
│   REPLACE area WITH area.area
│   SELECT (m.sel)
└ENDIF
```

```
*
*
*   _QSFOKPG47          m.obj VALID
*
*   Function Origin:
*
*   From Platform:      Windows
*   From Screen:        OBJECT,     Record Number:    8
*   Variable:           m.obj
*   Called By:          VALID Clause
*   Object Type:        Push Button
*   Snippet Number:     4
*
*
*
FUNCTION _qsf0kpg47     &&  m.obj VALID
#REGION 1
DO CASE
CASE m.object = 1
   SELECT dict
CASE m.object = 2
   SELECT disease
ENDCASE
=creatarray()
DO selitem
SHOW GETS


*
*
*   _QSFOKPGOJ          m.object VALID
*
*   Function Origin:
*
*   From Platform:      MS-DOS
*   From Screen:        OBJECT,     Record Number:    13
*   Variable:           m.object
*   Called By:          VALID Clause
*   Object Type:        Radio Button
*   Snippet Number:     5
*
*
*
FUNCTION _qsf0kpgoj     &&  m.object VALID
#REGION 1
IF m.object = 1
   SELECT dict
   SET ORDER TO 1
   GOTO BOTTOM
   m.id = id + 1
ELSE
   IF m.object = 2
      SELECT disease
      SET ORDER TO 1
      GOTO BOTTOM
      m.id = id + 1
   ENDIF
ENDIF
SHOW GETS

*
*
*   _QSFOKPGSR          m.name VALID
*
*   Function Origin:
*
*   From Platform:      MS-DOS
*   From Screen:        OBJECT,     Record Number:    14
*   Variable:           m.name
*   Called By:          VALID Clause
*   Object Type:        Field
*   Snippet Number:     6
*
*
*
FUNCTION _qsf0kpgsr     &&  m.name VALID
#REGION 1
=getobj(m.name)

*
*
*   _QSFOKPGVZ          m.button VALID
*
*   Function Origin:
*
*   From Platform:      MS-DOS
*   From Screen:        OBJECT,     Record Number:    16
*   Variable:           m.button
*   Called By:          VALID Clause
*   Object Type:        Push Button
*   Snippet Number:     7
*
*
*
FUNCTION _qsf0kpgvz     &&  m.button VALID
#REGION 1
IF m.button = 1
   * SEEK m.id
   * IF !found()
   APPEND BLANK
   REPLACE id WITH m.id
   REPLACE name WITH m.name
   *         IF !m.get
   *            = datainput()
   *         ENDIF
   * ENDIF
   DO CASE
   CASE m.status = "Q"
      SELECT quals
   CASE m.status = "G"
      SELECT goals
   ENDCASE
   APPEND BLANK
   REPLACE id WITH m.id
   REPLACE OBJECT WITH IIF(m.object=1,"S","D")
   REPLACE area WITH area.area
   SELECT (m.sel)
ENDIF

*
*
*   _QSFOKPH17          m.obj VALID
*
*   Function Origin:
*
*   From Platform:      MS-DOS
*   From Screen:        OBJECT,     Record Number:    18
*   Variable:           m.obj
*   Called By:          VALID Clause
*   Object Type:        Push Button
*   Snippet Number:     8
*
```

```
*
*
FUNCTION _qsf0kph17     &&  m.obj VALID
#REGION 1
DO CASE
CASE m.object = 1
   SELECT dict
CASE m.object = 2
   SELECT disease
ENDCASE
=creatarray()
DO selitem
SHOW GETS



*
*
*          OBJECT/MS-DOS Supporting Procedures and Functions
*
*
*

#REGION 1
FUNCTION creatarray
IF RECCOUNT() > 0
   DIMENSION marray[reccount()]
ELSE
   DIMENSION marray[1]
ENDIF
marray = ""
i = 0
IF m.status = "Q"
   m.dbf = 'QUALS'
ELSE
   m.dbf = 'GOALS'
ENDIF
GOTO TOP
SCAN
   IF EMPTY(SEEK(id,m.dbf))
      i = i + 1
      marray[i] = RIGHT(STR(id),5) + "  " + TRIM(name)
   ENDIF
ENDSCAN
IF i > 0
   DECLARE marray[i] && resize marray
ELSE
   DECLARE marray[1]
   marray = ""
ENDIF
RETURN

FUNCTION datainput
m.get = .T.
DO CASE
CASE "DISEASE.DBF" $ DBF()
   DO disease.spr
CASE "DICT.DBF" $ DBF()
   DO dict.spr WITH m.id
ENDCASE
m.id = id
m.name = name
RETURN

*-----------------------------------------------
*
*  Get the matching list of object
*
*-----------------------------------------------

FUNCTION getobj
PARAMETER m.name
PRIVATE m.file, mcom, msel,m.temp
mcom = SET("EXACT")
SET EXACT OFF
m.file = DBF()
msel = SELECT()
m.temp = m.name + "%"

SELECT id, name;
   FROM (m.file);
   INTO CURSOR t0000000;
   WHERE UPPER(name) LIKE (UPPER(m.temp))
SELECT t0000000
=creatarray()
USE
SELECT (msel)
DO selitem
*delete file t0000000.dbf
SET EXACT &mcom
RETURN


FUNCTION selitem
m.item = ""
IF !EMPTY(marray[1])
   DO oblist.spr WITH marray, m.item
   IF m.item $ "~"
      m.name = ""
      _CUROBJ = OBJNUM(m.name)
      SHOW GETS
      RETURN
   ENDIF
ENDIF
IF EMPTY(m.item)
   m.get = .T.
   DO CASE
   CASE "DISEASE.DBF" $ DBF()
      DO disease.spr
   CASE "DICT.DBF" $ DBF()
      DO dict.spr WITH m.id
   ENDCASE
   m.id = id
   m.name = name
ELSE
   m.id = VAL(m.item)
   m.name = TRIM(SUBSTR(m.item,7,LEN(m.item)))
ENDIF
*:  EOF: OBJECT.ac1
```

```
*
*
*     08/09/94            ENUM.SPR               09:39:43
*
*
*     Author's Name
*
*     Copyright (c) 1994 Company Name
*     Address
*     City,     Zip
*
*     Description:
*     This program was automatically generated by GENSCRN.
*
*

DO CASE
CASE _WINDOWS


   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
   ENDIF
   m.compstat = SET("COMPATIBLE")
   SET COMPATIBLE FOXPLUS

   m.rborder = SET("READBORDER")
   SET readborder ON

   *
   *
   *                  Windows Window definitions
   *
   *
   *

   IF NOT WEXIST("w_qenum") ;
         OR UPPER(WTITLE("W_QENUM")) == "W_QENUM.PJX" ;
         OR UPPER(WTITLE("W_QENUM")) == "W_QENUM.SCX" ;
         OR UPPER(WTITLE("W_QENUM")) == "W_QENUM.MNX" ;
         OR UPPER(WTITLE("W_QENUM")) == "W_QENUM.PRG" ;
         OR UPPER(WTITLE("W_QENUM")) == "W_QENUM.FRX" ;
         OR UPPER(WTITLE("W_QENUM")) == "W_QENUM.QPR"
      DEFINE WINDOW w_qenum ;
         AT  0.000, 0.000  ;
         SIZE 10.250,58.125 ;
         TITLE "Enumerated Type Editor" ;
         FONT "Terminal", 8 ;
         FLOAT ;
         CLOSE ;
         SHADOW ;
         NOMINIMIZE
      MOVE WINDOW w_qenum CENTER
   ENDIF


   *
   *
   *                  ENUM/Windows Screen Layout
   *
   *
   *

   #REGION 1
   IF WVISIBLE("w_qenum")
      ACTIVATE WINDOW w_qenum SAME
   ELSE
      ACTIVATE WINDOW w_qenum NOSHOW
   ENDIF
   @ 7.333,24.000 GET mbuttons ;
      PICTURE "@*HT OK;Cancel" ;
      SIZE 1.917,7.500,0.750 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kpjam()
   @ 4.583,1.625 SAY "Ord"  ;
      FONT "Terminal", 8
   @ 0.583,1.500 SAY "Mutex"  ;
      FONT "Terminal", 8
   @ 4.583,7.250 GET enum.ord ;
      SIZE 1.000,2.000 ;
      DEFAULT " " ;
      FONT "Terminal", 8 ;
      DISABLE
   @ 0.750,7.500 GET enum.mutex ;
      SIZE 1.000,1.000 ;
      DEFAULT " " ;
      FONT "Terminal", 8 ;
      VALID _qsf0kpjf7()
   @ 2.500,7.500 GET enum.enumerate ;
      SIZE 1.333,48.125 ;
      DEFAULT " " ;
      FONT "Terminal", 8 ;
      PICTURE "@!"
   @ 2.583,1.500 SAY "Enum"  ;
      FONT "Terminal", 8
   @ 7.250,17.375 GET mbbutton ;
      PICTURE "@*VN Add" ;
      SIZE 1.917,6.000,1.083 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kpjjk()

   IF NOT WVISIBLE("w_qenum")
      ACTIVATE WINDOW w_qenum
   ENDIF

   READ CYCLE MODAL

   RELEASE WINDOW w_qenum

   #REGION 0

   SET readborder &rborder

```

```
     IF m.talkstat = "ON"
          SET TALK ON
     ENDIF
     IF m.compstat = "ON"
          SET COMPATIBLE ON
     ENDIF


CASE _DOS


     #REGION 0
     REGIONAL m.currarea, m.talkstat, m.compstat

     IF SET("TALK") = "ON"
          SET TALK OFF
          m.talkstat = "ON"
     ELSE
          m.talkstat = "OFF"
     ENDIF
     m.compstat = SET("COMPATIBLE")
     SET COMPATIBLE FOXPLUS

     *
     *
     *                    MS-DOS Window definitions
     *
     *
     *

     IF NOT WEXIST("w_qenum") ;
               OR UPPER(WTITLE("W_QENUM")) == "W_QENUM.PJX" ;
               OR UPPER(WTITLE("W_QENUM")) == "W_QENUM.SCX" ;
               OR UPPER(WTITLE("W_QENUM")) == "W_QENUM.MNX" ;
               OR UPPER(WTITLE("W_QENUM")) == "W_QENUM.PRG" ;
               OR UPPER(WTITLE("W_QENUM")) == "W_QENUM.FRX" ;
               OR UPPER(WTITLE("W_QENUM")) == "W_QENUM.QPR"
          DEFINE WINDOW w_qenum ;
               FROM INT((SROW()-6)/2),INT((SCOL()-71)/2) ;
               TO INT((SROW()-6)/2)+5,INT((SCOL()-71)/2)+70 ;
               TITLE "Enumerated Type Editor" ;
               FLOAT ;
               CLOSE ;
               SHADOW ;
               NOMINIMIZE ;
               COLOR SCHEME 1
     ENDIF


     *
     *
     *                    ENUM/MS-DOS Screen Layout
     *
     *
     *

     #REGION 1
     IF WVISIBLE("w_qenum")
          ACTIVATE WINDOW w_qenum SAME
     ELSE
          ACTIVATE WINDOW w_qenum NOSHOW
     ENDIF
     @ 3,32 GET mbuttons ;
          PICTURE "@*HT OK;Cancel" ;
          SIZE 1,8,1 ;
          DEFAULT 1 ;
          VALID _qsf0kpjyy()
     @ 2,0 SAY "Ord" ;
          SIZE 1,3, 0
     @ 0,0 SAY "Mutex" ;
          SIZE 1,5, 0
     @ 2,6 GET enum.ord ;
          SIZE 1,2 ;
          DEFAULT " " ;
          DISABLE
     @ 0,6 GET enum.mutex ;
          SIZE 1,1 ;
          DEFAULT " " ;
          VALID _qsf0kpk3s()
     @ 1,6 GET enum.enumerate ;
          SIZE 1,63 ;
          DEFAULT " " ;
          PICTURE "@!"
     @ 1,0 SAY "Enum" ;
          SIZE 1,4, 0
     @ 3,22 GET mbbutton ;
          PICTURE "@*VN Add" ;
          SIZE 1,8,1 ;
          DEFAULT 1 ;
          VALID _qsf0kpk8e()

     IF NOT WVISIBLE("w_qenum")
          ACTIVATE WINDOW w_qenum
     ENDIF

     READ CYCLE MODAL

     RELEASE WINDOW w_qenum

     #REGION 0
     IF m.talkstat = "ON"
          SET TALK ON
     ENDIF
     IF m.compstat = "ON"
          SET COMPATIBLE ON
     ENDIF


ENDCASE


     *
     *
     *         _QSFOKPJAM          mbuttons VALID
     *
     *         Function Origin:
     *
     *         From Platform:      Windows
     *         From Screen:        ENUM,     Record Number:    2
     *         Variable:           mbuttons
     *         Called By:          VALID Clause
     *         Object Type:        Push Button
*       Snippet Number:      1
*
*
*
FUNCTION _qsf0kpjam     &&  mbuttons VALID
#REGION 1
┌DO CASE
├CASE mbuttons = 1   && ok
   mok = .T.
├CASE mbuttons = 2   && cancel
   mok = .F.
└ENDCASE

*
*
*       _QSFOKPJF7           enum.mutex VALID
*
*       Function Origin:
*
*       From Platform:       Windows
*       From Screen:         ENUM,     Record Number:    6
*       Variable:            enum.mutex
*       Called By:           VALID Clause
*       Object Type:         Field
*       Snippet Number:      2
*
*
*
FUNCTION _qsf0kpjf7     &&  enum.mutex VALID
#REGION 1
 ┌DO CASE
 ├CASE dict.datatype $ "EL"
<═════RETURN mutex = "-"
 ├CASE dict.datatype $ "M"
<═════RETURN mutex $ "-+"
 └ENDCASE
<══RETURN .F.

*
*
*       _QSFOKPJJK           mbbutton VALID
*
*       Function Origin:
*
*       From Platform:       Windows
*       From Screen:         ENUM,     Record Number:    9
*       Variable:            mbbutton
*       Called By:           VALID Clause
*       Object Type:         Push Button
*       Snippet Number:      3
*
*
*
FUNCTION _qsf0kpjjk     &&  mbbutton VALID
#REGION 1
m.ord = 0
┌DO WHILE id = dict.id
│  m.ord = ord
│  SKIP
└ENDDO
m.ord  =  m.ord + 1
APPEND BLANK
m.id  =  dict.id
m.mutex  =  "-"
GATHER MEMVAR
_CUROBJ = OBJNUM(enum.mutex)
SHOW GETS


*
*
*       _QSFOKPJYY           mbuttons VALID
*
*       Function Origin:
*
*       From Platform:       MS-DOS
*       From Screen:         ENUM,     Record Number:   12
*       Variable:            mbuttons
*       Called By:           VALID Clause
*       Object Type:         Push Button
*       Snippet Number:      4
*
*
*
FUNCTION _qsf0kpjyy     &&  mbuttons VALID
#REGION 1
┌DO CASE
├CASE mbuttons = 1   && ok
   mok = .T.
├CASE mbuttons = 2   && cancel
   mok = .F.
└ENDCASE

*
*
*       _QSFOKPK3S           enum.mutex VALID
*
*       Function Origin:
*
*       From Platform:       MS-DOS
*       From Screen:         ENUM,     Record Number:   16
*       Variable:            enum.mutex
*       Called By:           VALID Clause
*       Object Type:         Field
*       Snippet Number:      5
*
*
*
FUNCTION _qsf0kpk3s     &&  enum.mutex VALID
#REGION 1
┌DO CASE
├CASE dict.datatype $ "EL"
<═════RETURN mutex = "-"
├CASE dict.datatype $ "M"
<═════RETURN mutex $ "-+"
└ENDCASE
<══RETURN .F.

*
*
*       _QSFOKPK8E           mbbutton VALID
*
*       Function Origin:
*
*       From Platform:       MS-DOS
*       From Screen:         ENUM,     Record Number:   19
*       Variable:            mbbutton
*       Called By:           VALID Clause
*       Object Type:         Push Button
*       Snippet Number:      6
*
*
```

```
*
FUNCTION _qsf0kpk8e     &&  mbbutton VALID
#REGION 1
m.ord = 0
DO WHILE id = dict.id
   m.ord = ord
   SKIP
ENDDO
m.ord  =  m.ord + 1
APPEND BLANK
m.id  =  dict.id
m.mutex  =  "-"
GATHER MEMVAR
 _CUROBJ = OBJNUM(enum.mutex)
SHOW GETS
*:  EOF: ENUM.ac1
```

```
*
*
*       08/09/94              RESTORE.SPR              09:39:47
*
*
*       Author's Name
*
*       Copyright (c) 1994 Company Name
*       Address
*       City,     Zip
*
*       Description:
*       This program was automatically generated by GENSCRN.
*
*

PARAMETERS MESSAGE, wildcard, filename
DO CASE
CASE _WINDOWS


   *
   *
   *        RESTORE/Windows Setup Code - SECTION 1
   *
   *
   *

   #REGION 1
   PRIVATE mfanme, mpath, olddrive, predrive, oldpath, maction, mdrive, mfiles
   DO CASE
   CASE PARAMETERS() = 0
      m.message = ""
      m.wildcard = "*.zip"
      mfname = ""
   CASE PARAMETERS() = 1 OR EMPTY(m.wildcard)
      m.wildcard = "*.zip"
      mfname = ""
   CASE PARAMETERS() = 2 OR EMPTY(m.filename)
      mfname = ""
   OTHERWISE
      mfname = filename
   ENDCASE


   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
   ENDIF
   m.compstat = SET("COMPATIBLE")
   SET COMPATIBLE FOXPLUS

   m.rborder = SET("READBORDER")
   SET readborder ON

   *
   *
   *        Windows Window definitions
   *
   *
   *

   IF NOT WEXIST("w_restore") ;
         OR UPPER(WTITLE("W_RESTORE")) == "W_RESTORE.PJX" ;
         OR UPPER(WTITLE("W_RESTORE")) == "W_RESTORE.SCX" ;
         OR UPPER(WTITLE("W_RESTORE")) == "W_RESTORE.MNX" ;
         OR UPPER(WTITLE("W_RESTORE")) == "W_RESTORE.PRG" ;
         OR UPPER(WTITLE("W_RESTORE")) == "W_RESTORE.FRX" ;
         OR UPPER(WTITLE("W_RESTORE")) == "W_RESTORE.QPR"
      DEFINE WINDOW w_restore ;
         AT  0.000, 0.000  ;
         SIZE 18.846,79.500 ;
         FONT "MS Sans Serif", 8 ;
         STYLE "B" ;
         NOFLOAT ;
         NOCLOSE ;
         SHADOW ;
         NOMINIMIZE ;
         DOUBLE
      MOVE WINDOW w_restore CENTER
   ENDIF


   *
   *
   *        RESTORE/Windows Setup Code - SECTION 2
   *
   *
   *

   #REGION 1
   m.olddrive = SET("DEFAULT")
   m.prevdrive = m.olddrive
   m.oldpath = CURDIR()
   maction = 1

   DECLARE drivearray[1,1]
   mdrive = 1
   DO newdrivepop

   DECLARE patharray[1,1]
   mpath = 1
   DO newpathpop

   DECLARE filearray[1,1]
   mfiles = 1
   DO newfilepop


   *
   *
   *      RESTORE/Windows Screen Layout
   *
   *
   *

   #REGION 1
   IF WVISIBLE("w_restore")
      ACTIVATE WINDOW w_restore SAME
   ELSE
      ACTIVATE WINDOW w_restore NOSHOW
   ENDIF
   @ 3.692,59.500 GET mdrive ;
      PICTURE "@^" ;
      FROM drivearray ;
      SIZE 1.500,12.500 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      WHEN _qsf0kpmpz() ;
      VALID _qsf0kpms0()
   @ 7.154,42.667 SAY "Directory:"  ;
      FONT "Terminal", 8
   @ 0.615,3.167 SAY MESSAGE ;
      SIZE 1.000,39.875 ;
      FONT "Terminal", 8
   @ 7.000,59.500 GET mpath ;
      PICTURE "@^" ;
      FROM patharray ;
      SIZE 1.500,12.500 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kpmxi()
   @ 10.231,43.333 GET maction ;
      PICTURE "@*HT \!\<Restore;\<Cancel" ;
      SIZE 2.250,11.375,1.625 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kpn0j()
   @ 3.154,2.833 GET mfile ;
      PICTURE "@&N" ;
      FROM filearray ;
      SIZE 11.667,28.875 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      VALID _qsf0kpn3t()
   @ 15.846,29.000 GET mfname ;
      SIZE 1.000,32.500 ;
      DEFAULT " " ;
      FONT "Terminal", 8 ;
      VALID _qsf0kpn7e()
   @ 15.846,3.167 SAY "Restore file name: "  ;
      FONT "Terminal", 8
   @ 3.923,42.667 SAY "From drive: "  ;
      FONT "Terminal", 8

   IF NOT WVISIBLE("w_restore")
      ACTIVATE WINDOW w_restore
   ENDIF

   READ CYCLE MODAL

   RELEASE WINDOW w_restore

   #REGION 0

   SET readborder &rborder

   IF m.talkstat = "ON"
      SET TALK ON
   ENDIF
   IF m.compstat = "ON"
      SET COMPATIBLE ON
   ENDIF


   *
   *
   *      RESTORE/Windows Cleanup Code
   *
   *
   *

   #REGION 1
   SET DEFAULT TO (m.olddrive + m.oldpath)
<--RETURN

   ********************************************************

   *********************

CASE _DOS


   *
   *
   *      RESTORE/MS-DOS Setup Code - SECTION 1
   *
   *
   *

   #REGION 1
   PRIVATE mfanme, mpath, olddrive, predrive, oldpath, maction, mdriv
=> e, mfiles
    DO CASE
    CASE PARAMETERS() = 0
       m.message = ""
       m.wildcard = "*.zip"
       mfname = ""
    CASE PARAMETERS() = 1 OR EMPTY(m.wildcard)
       m.wildcard = "*.zip"
       mfname = ""
    CASE PARAMETERS() = 2 OR EMPTY(m.filename)
       mfname = ""
    OTHERWISE
       mfname = filename
    ENDCASE
```

```
#REGION 0
REGIONAL m.currarea, m.talkstat, m.compstat

IF SET("TALK") = "ON"
   SET TALK OFF
   m.talkstat = "ON"
ELSE
   m.talkstat = "OFF"
ENDIF
m.compstat = SET("COMPATIBLE")
SET COMPATIBLE FOXPLUS

*
*
*               MS-DOS Window definitions
*
*
*

IF NOT WEXIST("w_restore") ;
      OR UPPER(WTITLE("W_RESTORE")) == "W_RESTORE.PJX" ;
      OR UPPER(WTITLE("W_RESTORE")) == "W_RESTORE.SCX" ;
      OR UPPER(WTITLE("W_RESTORE")) == "W_RESTORE.MNX" ;
      OR UPPER(WTITLE("W_RESTORE")) == "W_RESTORE.PRG" ;
      OR UPPER(WTITLE("W_RESTORE")) == "W_RESTORE.FRX" ;
      OR UPPER(WTITLE("W_RESTORE")) == "W_RESTORE.QPR"
   DEFINE WINDOW w_restore ;
      FROM INT((SROW()-18)/2),INT((SCOL()-61)/2) ;
      TO INT((SROW()-18)/2)+17,INT((SCOL()-61)/2)+60 ;
      NOFLOAT ;
      NOCLOSE ;
      SHADOW ;
      NOMINIMIZE ;
      DOUBLE ;
      COLOR SCHEME 5
ENDIF

*
*
*          RESTORE/MS-DOS Setup Code - SECTION 2
*
*
*

#REGION 1
m.olddrive = SET("DEFAULT")
m.prevdrive = m.olddrive
m.oldpath = CURDIR()
maction = 1

DECLARE drivearray[1,1]
mdrive = 1
DO newdrivepop

DECLARE patharray[1,1]
mpath = 1
DO newpathpop

DECLARE filearray[1,1]
mfiles = 1
DO newfilepop


*
*
*               RESTORE/MS-DOS Screen Layout
*
*
*

#REGION 1
IF WVISIBLE("w_restore")
   ACTIVATE WINDOW w_restore SAME
ELSE
   ACTIVATE WINDOW w_restore NOSHOW
ENDIF
@ 2,40 GET mdrive ;
   PICTURE "@^" ;
   FROM drivearray ;
   SIZE 3,18 ;
   DEFAULT 1 ;
   WHEN _qsf0kpny5() ;
   VALID _qsf0kpo0a() ;
   COLOR SCHEME 5, 6
@ 6,28 SAY "Directory:" ;
   SIZE 1,10, 0
@ 0,0 SAY MESSAGE ;
   SIZE 1,40
@ 5,40 GET mpath ;
   PICTURE "@^" ;
   FROM patharray ;
   SIZE 3,18 ;
   DEFAULT 1 ;
   VALID _qsf0kpo5t() ;
   COLOR SCHEME 5, 6
@ 9,45 GET maction ;
   PICTURE "@*VT \!\<Restore;\<Cancel" ;
   SIZE 1,9,1 ;
   DEFAULT 1 ;
   VALID _qsf0kpo8t()
@ 2,1 GET mfile ;
   PICTURE "@&N" ;
   FROM filearray ;
   SIZE 11,26 ;
   DEFAULT 1 ;
   VALID _qsf0kpoby() ;
   COLOR SCHEME 6
@ 14,19 GET mfname ;
   SIZE 1,39 ;
   DEFAULT " " ;
   VALID _qsf0kpofg()
@ 14,0 SAY "Restore file name: " ;
   SIZE 1,19, 0
@ 3,28 SAY "From drive: " ;
   SIZE 1,12, 0
```

```

  ┌IF NOT WVISIBLE("w_restore")
  │   ACTIVATE WINDOW w_restore
  └ENDIF

   READ CYCLE MODAL

   RELEASE WINDOW w_restore

   #REGION 0
  ┌IF m.talkstat = "ON"
  │   SET TALK ON
  └ENDIF
  ┌IF m.compstat = "ON"
  │   SET COMPATIBLE ON
  └ENDIF


    *     ┌──────────────────────────────────────────
    *     │
    *     │         RESTORE/MS-DOS Cleanup Code
    *     │
    *     └──────────────────────────────────────────
    *

    #REGION 1
    SET DEFAULT TO (m.olddrive + m.oldpath)
 <──RETURN

    ******************************************************

    *********************

 └ENDCASE


 *
 *      ┌──────────────────────────────────────────────────
 *      │ _QSFOKPMPZ            mdrive WHEN
 *      │
 *      │ Function Origin:
 *      │
 *      │ From Platform:        Windows
 *      │ From Screen:          RESTORE,     Record Number:    2
 *      │ Variable:             mdrive
 *      │ Called By:            WHEN Clause
 *      │ Object Type:          Popup
 *      │ Snippet Number:       1
 *      │
 *      └──────────────────────────────────────────────────
 *
 FUNCTION _qsf0kpmpz     &&  mdrive WHEN
 #REGION 1
 m.prevdrive = mdrive

 *
 *      ┌──────────────────────────────────────────────────
 *      │ _QSFOKPMSO            mdrive VALID
 *      │
 *      │ Function Origin:
 *      │
 *      │ From Platform:        Windows
 *      │ From Screen:          RESTORE,     Record Number:    2
 *      │ Variable:             mdrive
 *      │ Called By:            VALID Clause
 *      │ Object Type:          Popup
 *      │ Snippet Number:       2
 *      │
 *      └──────────────────────────────────────────────────
 *
 *Switch to the selected drive
 FUNCTION _qsf0kpms0     &&  mdrive VALID
 #REGION 1
 PRIVATE newdrive,mready

 *Convert the popup bar number into the matching drive name
 m.newdrive = drivearray[mdrive]

┌IF UPPER(m.newdrive) $ "A:B:"
│   mready = yesno("Please insert disk into drive " + m.newdrive, "ready","cancel")
├ELSE
│   mready = .T.
└ENDIF
┌IF mready
│   *Go there and reset all the other popups to match
│   SET DEFAULT TO (m.newdrive)
│   DO newpathpop
│   DO newfilepop
├ELSE
│   mdrive =  m.prevdrive
│   m.newdrive = drivearray[mdrive]
└ENDIF
 SHOW GETS

 *
 *      ┌──────────────────────────────────────────────────
 *      │ _QSFOKPMXI            mpath VALID
 *      │
 *      │ Function Origin:
 *      │
 *      │ From Platform:        Windows
 *      │ From Screen:          RESTORE,     Record Number:    5
 *      │ Variable:             mpath
 *      │ Called By:            VALID Clause
 *      │ Object Type:          Popup
 *      │ Snippet Number:       3
 *      │
 *      └──────────────────────────────────────────────────
 *
 FUNCTION _qsf0kpmxi     &&  mpath VALID
 #REGION 1
 m.newdefault = pathstring()
 SET DEFAULT TO (m,newdefault)
 DO newpathpop
 DO newfilepop
 SHOW GETS

 *
 *      ┌──────────────────────────────────────────────────
 *      │ _QSFOKPNOJ            maction VALID
 *      │
 *      │ Function Origin:
 *      │
 *      │ From Platform:        Windows
 *      │ From Screen:          RESTORE,     Record Number:    6
 *      │ Variable:             maction
*        │ Called By:           VALID Clause
*        │ Object Type:         Push Button
*        │ Snippet Number:      4
*        └─────────────────────────────────────────
*
FUNCTION _qsf0kpn0j      &&  maction VALID
#REGION 1
┌DO CASE
├CASE maction = 1
│  ┌IF FILE(mfname)
│  │   DO pkunzip
│  └ENDIF
├OTHERWISE
└ENDCASE

*
*        ┌─────────────────────────────────────────
*        │ _QSF0KPN3T           mfile VALID
*        │
*        │ Function Origin:
*        │
*        │ From Platform:       Windows
*        │ From Screen:         RESTORE,     Record Number:    7
*        │ Variable:            mfile
*        │ Called By:           VALID Clause
*        │ Object Type:         List
*        │ Snippet Number:      5
*        └─────────────────────────────────────────
*
*
FUNCTION _qsf0kpn3t      &&  mfile VALID
#REGION 1
mnewfile = filearray[mfile,1]
┌IF "[" $ mnewfile
│   mnewpath = SUBSTR(mnewfile,2,LEN(mnewfile)-2)
│   SET DEFA TO (mnewpath)
│   DO newpathpop
│   DO newfilepop
│   SHOW GETS
├ELSE
│   mfname = mnewfile
│   SHOW GETS
└ENDIF

*
*        ┌─────────────────────────────────────────
*        │ _QSF0KPN7E           mfname VALID
*        │
*        │ Function Origin:
*        │
*        │ From Platform:       Windows
*        │ From Screen:         RESTORE,     Record Number:    8
*        │ Variable:            mfname
*        │ Called By:           VALID Clause
*        │ Object Type:         Field
*        │ Snippet Number:      6
*        └─────────────────────────────────────────
*
*
FUNCTION _qsf0kpn7e      &&  mfname VALID
#REGION 1
┌IF !EMPTY(mfname)
│  ┌IF !FILE(mfname)
│  │   mfname = ""
│  └ENDIF
└ENDIF

*
*
*        ┌─────────────────────────────────────────
*        │ _QSF0KPNY5           mdrive WHEN
*        │
*        │ Function Origin:
*        │
*        │ From Platform:       MS-DOS
*        │ From Screen:         RESTORE,     Record Number:    14
*        │ Variable:            mdrive
*        │ Called By:           WHEN Clause
*        │ Object Type:         Popup
*        │ Snippet Number:      7
*        └─────────────────────────────────────────
*
*
FUNCTION _qsf0kpny5      &&  mdrive WHEN
#REGION 1
m.prevdrive = mdrive

*
*        ┌─────────────────────────────────────────
*        │ _QSF0KPO0A           mdrive VALID
*        │
*        │ Function Origin:
*        │
*        │ From Platform:       MS-DOS
*        │ From Screen:         RESTORE,     Record Number:    14
*        │ Variable:            mdrive
*        │ Called By:           VALID Clause
*        │ Object Type:         Popup
*        │ Snippet Number:      8
*        └─────────────────────────────────────────
*
*
*Switch to the selected drive
FUNCTION _qsf0kpo0a      &&  mdrive VALID
#REGION 1
PRIVATE newdrive,mready

*Convert the popup bar number into the matching drive name
m.newdrive = drivearray[mdrive]

┌IF UPPER(m.newdrive) $ "A:B:"
│   mready = yesno("Please insert disk into drive " + m.newdrive, "rea
=> dy","cancel")
├ELSE
│   mready = .T.
└ENDIF
┌IF mready
│   *Go there and reset all the other popups to match
│   SET DEFAULT TO (m.newdrive)
│   DO newpathpop
│   DO newfilepop
├ELSE
│   mdrive =  m.prevdrive
│   m.newdrive = drivearray[mdrive]
└ENDIF
SHOW GETS

*
*        ┌─────────────────────────────────────────
*        │ _QSF0KPO5T           mpath VALID
*        │
*        │ Function Origin:
    *
    *       From Platform:      MS-DOS
    *       From Screen:        RESTORE,      Record Number:   17
    *       Variable:           mpath
    *       Called By:          VALID Clause
    *       Object Type:        Popup
    *       Snippet Number:     9
    *
    *
    *
    FUNCTION _qsf0kpo5t     &&  mpath VALID
    #REGION 1
    m.newdefault = pathstring()
    SET DEFAULT TO (m.newdefault)
    DO newpathpop
    DO newfilepop
    SHOW GETS

    *
    *
    *       _QSFOKPO8T          maction VALID
    *
    *       Function Origin:
    *
    *       From Platform:      MS-DOS
    *       From Screen:        RESTORE,      Record Number:   18
    *       Variable:           maction
    *       Called By:          VALID Clause
    *       Object Type:        Push Button
    *       Snippet Number:     10
    *
    *
    *
    FUNCTION _qsf0kpo8t     &&  maction VALID
    #REGION 1
   ┌DO CASE
   ├CASE maction = 1
   │  ┌IF FILE(mfname)
   │  │   DO pkunzip
   │  └ENDIF
   ├OTHERWISE
   └ENDCASE

    *
    *
    *       _QSFOKPOBY          mfile VALID
    *
    *       Function Origin:
    *
    *       From Platform:      MS-DOS
    *       From Screen:        RESTORE,'     Record Number:   19
    *       Variable:           mfile
    *       Called By:          VALID Clause
    *       Object Type:        List
    *       Snippet Number:     11
    *
    *
    *
    FUNCTION _qsf0kpoby     &&  mfile VALID
    #REGION 1
    mnewfile = filearray[mfile,1]
   ┌IF "[" $ mnewfile
   │  mnewpath = SUBSTR(mnewfile,2,LEN(mnewfile)-2)
   │  SET DEFA TO (mnewpath)
   │  DO newpathpop
   │  DO newfilepop
   │  SHOW GETS
   ├ELSE
   │  mfname = mnewfile
   │  SHOW GETS
   └ENDIF

    *
    *
    *       _QSFOKPOFG          mfname VALID
    *
    *       Function Origin:
    *
    *       From Platform:      MS-DOS
    *       From Screen:        RESTORE,      Record Number:   20
    *       Variable:           mfname
    *       Called By:          VALID Clause
    *       Object Type:        Field
    *       Snippet Number:     12
    *
    *
    *
    FUNCTION _qsf0kpofg     &&  mfname VALID
    #REGION 1
   ┌IF !EMPTY(mfname)
   │  ┌IF !FILE(mfname)
   │  │   mfname = ""
   │  └ENDIF
   └ENDIF


    *
    *
    *       RESTORE/MS-DOS Supporting Procedures and Functions
    *
    *
    *

    #REGION 1

    *
    *
    *       RESTORE Procedure NEWDRIVEPOP
    *
    *
    *


    PROCEDURE newdrivepop
   ┌DO CASE
   ├CASE  DOS
   │  *********************
   │  * Create a popup with each of the legal drive names
   │  * The popup will be based on an array of the drive names
   │  PRIVATE olddefault, olderror, drivename, ERROR
   │
   │  * We will change drives, so remember where we started
   │  m.olddefault = SET( "DEFAULT" )
   │
   │  * To test for legal drives this program SET DEFAULT TO each drive
   │  * If no error occurs the drive name will be added to an array
   │
   │  * Create the error trap
   │  m.olderror = ON( "ERROR" )
   │  m.error = .F.
   │  ON ERROR m.error = .T.
   │
```

IC/14PRINT

```
   * Create the array of legal drive names
   DECLARE drivearray[ 1 ]
   drivearray[ 1 ] = ""

   * Loop from A to Z
   m.drivename = "A"
   FOR i = 1 TO 26

      * Try to switch to the drive
      SET DEFAULT TO (m.drivename)

      * If it worked
      IF NOT m.error

         * Add the name to the last element in the array
         drivearray[ ALEN( DriveArray ) ] = m.drivename + ":"

         * Add another element at the end of the array
         DECLARE drivearray[ ALEN( DriveArray ) + 1 ]

      ENDIF

      * Change to the next letter in the alphabet
      m.drivename = CHR( ASC( m.drivename ) + 1 )

      * Reset the error trap
      m.error = .F.
   NEXT

   * If the first element is empty, no drives were found
   IF EMPTY( drivearray[ 1 ] )
      SHOW GET m.drive disabled
   ELSE
      * Drives were found so cut off the last empty element
      * added to the array in the loop
      DECLARE drivearray[ ALEN( DriveArray ) - 1 ]
   ENDIF

   * Reset the error trap and return to home
   ON ERROR &olderror
   SET DEFAULT TO (m.olddefault)

   * Initialize the popup variable to the element in the
   * drive array which contains the current drive
   mdrive = ASCAN( drivearray, m.olddefault )
<==RETURN

   ********************
CASE  WINDOWS
   ********************
   * Create a popup with each of the legal drive names
   * The popup will be based on an array of the drive names
   PRIVATE olddefault, olderror, drivename, ERROR

   * We will change drives, so remember where we started
   m.olddefault = SET( "DEFAULT" )

   * To test for legal drives this program SET DEFAULT TO each drive
   * If no error occurs the drive name will be added to an array

   * Create the error trap
   m.olderror = ON( "ERROR" )
   m.error = .F.
   ON ERROR m.error = .T.

   * Create the array of legal drive names
   DECLARE drivearray[ 3 ]
   drivearray[ 1 ] = "A"
   drivearray[ 2 ] = "B"
   drivearray[ 3 ] = ""

   * Loop from A to Z
   m.drivename = "C"
   FOR i = 3 TO 26

      * Try to switch to the drive
      SET DEFAULT TO (m.drivename)

      * If it worked
      IF NOT m.error

         * Add the name to the last element in the array
         drivearray[ ALEN( DriveArray ) ] = m.drivename + ":"

         * Add another element at the end of the array
         DECLARE drivearray[ ALEN( DriveArray ) + 1 ]

      ENDIF

      * Change to the next letter in the alphabet
      m.drivename = CHR( ASC( m.drivename ) + 1 )

      * Reset the error trap
      m.error = .F.
   NEXT

   * If the first element is empty, no drives were found
   IF EMPTY( drivearray[ 1 ] )
      SHOW GET m.drive disabled
   ELSE
      * Drives were found so cut off the last empty element
      * added to the array in the loop
      DECLARE drivearray[ ALEN( DriveArray ) - 1 ]
   ENDIF

   * Reset the error trap and return to home
   ON ERROR &olderror
   SET DEFAULT TO (m.olddefault)

   * Initialize the popup variable to the element in the
   * drive array which contains the current drive
   mdrive = ASCAN( drivearray, m.olddefault )
<==RETURN

   ********************
ENDCASE

*
*
*                RESTORE Procedure NEWPATHPOP
*
*
*


PROCEDURE newpathpop
DO CASE
CASE  DOS
   ********************
   * Create a popup with one bar for each subdirectory
   * in the current path
   PRIVATE dirstring, dircount
```

```

      * Base the popup on an array with one element for
      * each subdirectory in the current path
      *
      * Start the array with the current drive
      DECLARE patharray[ 1 ]
      patharray[ 1 ] = SET( "DEFAULT" )

      * Get the current path string
      m.dirstring = CURDIR()

      * If we are not in the root directory
     IF LEN( m.dirstring ) > 1

         * Start parsing the path string for each directory name
         m.dircount = 1

         * Continue as long as there is a pair of slashes
         * Surrounding the next part of the string
        DO WHILE AT( "\", m.dirstring, m.dircount ) <> 0 AND ;
               AT( "\", m.dirstring, m.dircount + 1 ) <> 0

            * Add another element at the end of the array
            DECLARE patharray[ m.DirCount + 1 ]

            * Cut out the next set of characters between the slashes
            m.firstchar = AT( "\", m.dirstring, m.dircount ) + 1

            m.pathlength = AT( "\", m.dirstring, m.dircount + 1 )  ;
               - m.firstchar

            * And add them to the next array element
            patharray[ m.DirCount + 1 ] = SUBSTR( m.dirstring, ;
               m.firstchar, m.pathlength )

            * Look for the next directory name in the path string
            m.dircount = m.dircount + 1

        ENDDO
     ENDIF

      * Point the popup variable at the last element in the array
      mpath = ALEN( patharray )
 <----RETURN

      ********************
   CASE _WINDOWS
      ********************
      * Create a popup with one bar for each subdirectory
      * in the current path
      PRIVATE dirstring, dircount

      * Base the popup on an array with one element for
      * each subdirectory in the current path
      *
      * Start the array with the current drive
      DECLARE patharray[ 1 ]
      patharray[ 1 ] = SET( "DEFAULT" )

      * Get the current path string
      m.dirstring = CURDIR()

      * If we are not in the root directory
     IF LEN( m.dirstring ) > 1

         * Start parsing the path string for each directory name
         m.dircount = 1

         * Continue as long as there is a pair of slashes
         * Surrounding the next part of the string
        DO WHILE AT( "\", m.dirstring, m.dircount ) <> 0 AND ;
               AT( "\", m.dirstring, m.dircount + 1 ) <> 0

            * Add another element at the end of the array
            DECLARE patharray[ m.DirCount + 1 ]

            * Cut out the next set of characters between the slashes
            m.firstchar = AT( "\", m.dirstring, m.dircount ) + 1

            m.pathlength = AT( "\", m.dirstring, m.dircount + 1 )  ;
               - m.firstchar

            * And add them to the next array element
            patharray[ m.DirCount + 1 ] = SUBSTR( m.dirstring, ;
               m.firstchar, m.pathlength )

            * Look for the next directory name in the path string
            m.dircount = m.dircount + 1

        ENDDO
     ENDIF

      * Point the popup variable at the last element in the array
      mpath = ALEN( patharray )
 <----RETURN

      ********************
   ENDCASE

   *
   *                    RESTORE Procedure NEWFILEPOP
   *
   *
   *
   *


   PROCEDURE newfilepop
   DO CASE
   CASE _DOS
      ********************

      * Create a popup with all of the file names and its subdirection i
=> n the current directory
      * This will be based on an array as well

      * Create an array with all the files matching the current wild car
=> d
      * ADIR() creates an array with 5 columns.

      PRIVATE startsort
      * Fill an array with directory names only
      =ADIR( dirarray, "", "D")
      SIZE = ALEN(dirarray,1)
     IF dirarray[ 1, 1 ] = "."
         =ADEL(dirarray,1)
         SIZE = SIZE - 1
         DECLARE dirarray[size,5]
     ELSE
         * We must be in the root directory "\"
         * Add one more row to the directory array
         SIZE = SIZE + 1
      DECLARE dirarray[ SIZE, 5 ]

      * Push all the rows down by one
      =AINS( dirarray, 1 )

      * Fill in the first directory name in the array
      * a bug in the popups makes it refuse to display a
      * prompt of "\", use "\\" to make one \ appear
      * even then the bar will automatically be non-selectable
      * which in this case is fine
      dirarray[ 1,1 ] = "\\"

      * Start the sort after the root name
   ENDIF

   IF ALEN( dirarray, 1 ) > 2

      * Sort the array starting at the starting row
      =ASORT( dirarray, AELEMENT( dirarray, 2, 1 ) )
   ENDIF
   FOR i = 1 TO ALEN(dirarray,1)
      dirarray[i,1] = "[" + dirarray[i,1] + "]"
   ENDFOR

   IF ADIR(filearray, m.wildcard) = 0
      SIZE = ALEN(dirarray,1)
      DECLARE filearray[size,5]
      =ACOPY(dirarray, filearray)
   ELSE
      stop = ALEN(dirarray,1)
      FOR i = 1 TO stop
         SIZE = ALEN(filearray,1)
         DECLARE filearray[size + 1,5]
         =AINS(filearray,1)
         filearray[1,1] = dirarray(alen[dirarray,1],1)
         IF ALEN(dirarray,1) > 1
            DECLARE dirarray[alen(dirarray,1)-1,5]
         ENDIF
      ENDFOR
   ENDIF
   mfile = 1
<--RETURN

   *******************
CASE  WINDOWS
   *******************

   * Create a popup with all of the file names and its subdirection i
=> n the current directory
   * This will be based on an array as well

   * Create an array with all the files matching the current wild car
=> d
   * ADIR() creates an array with 5 columns.

   PRIVATE startsort
   * Fill an array with directory names only
   =ADIR( dirarray, "", "D")
   SIZE = ALEN(dirarray,1)
   IF dirarray[ 1, 1 ] = "."
      =ADEL(dirarray,1)
      SIZE = SIZE - 1
      DECLARE dirarray[size,5]
   ELSE
      * We must be in the root directory "\"
      * Add one more row to the directory array
```

```
      SIZE = SIZE + 1
      DECLARE dirarray[ SIZE, 5 ]

      * Push all the rows down by one
      =AINS( dirarray, 1 )

      * Fill in the first directory name in the array
      * a bug in the popups makes it refuse to display a
      * prompt of "\", use "\\" to make one \ appear
      * even then the bar will automatically be non-selectable
      * which in this case is fine
      dirarray[ 1,1 ] = "\\"

      * Start the sort after the root name
   ENDIF

   IF ALEN( dirarray, 1 ) > 2

      * Sort the array starting at the starting row
      =ASORT( dirarray, AELEMENT( dirarray, 2, 1 ) )
   ENDIF
   FOR i = 1 TO ALEN(dirarray,1)
      dirarray[i,1] = "[" + dirarray[i,1] + "]"
   ENDFOR

   IF ADIR(filearray, m.wildcard) = 0
      SIZE = ALEN(dirarray,1)
      DECLARE filearray[size,5]
      =ACOPY(dirarray, filearray)
   ELSE
      stop = ALEN(dirarray,1)
      FOR i = 1 TO stop
         SIZE = ALEN(filearray,1)
         DECLARE filearray[size + 1,5]
         =AINS(filearray,1)
         filearray[1,1] = dirarray(alen[dirarray,1],1)
         IF ALEN(dirarray,1) > 1
            DECLARE dirarray[alen(dirarray,1)-1,5]
         ENDIF
      ENDFOR
   ENDIF
   mfile = 1
<--RETURN

   *******************
ENDCASE

 *
 *
 *            RESTORE Function PATHSTRING
 *
 *
 *


PROCEDURE pathstring
DO CASE
CASE  DOS
   *******************
   * Convert an array of subdirectories, such as PathArray,
   * into a legal path string for use with SET DEFAULT
   * Use the current value of the path popup as the
   * ending point on the path
   PRIVATE ppath

   * Start with the drive name
```

```
     m.ppath = patharray[ 1 ]

      * If the path popup is pointing to a subdirectory
    IF mpath > 1

        * Add all the subdirectories to the path string
       FOR i = 2 TO mpath
          m.ppath = m.ppath + "\" + patharray[ i ]

       NEXT
    ENDIF

     * End the path with one last slash for good luck
     m.ppath = m.ppath + "\"

<----RETURN m.ppath

     ********************
  CASE  WINDOWS
     ********************
     * Convert an array of subdirectories, such as PathArray,
     * into a legal path string for use with SET DEFAULT
     * Use the current value of the path popup as the
     * ending point on the path
     PRIVATE ppath

     * Start with the drive name
     m.ppath = patharray[ 1 ]

     * If the path popup is pointing to a subdirectory
    IF mpath > 1

        * Add all the subdirectories to the path string
       FOR i = 2 TO mpath
          m.ppath = m.ppath + "\" + patharray[ i ]

       NEXT
    ENDIF

     * End the path with one last slash for good luck
     m.ppath = m.ppath + "\"

<----RETURN m.ppath

     ********************
  ENDCASE

   *
   *
   *           RESTORE Procedure PKUNZIP
   *
   *
   *


   PROCEDURE pkunzip
  DO CASE
  CASE  DOS
     ********************

     PRIVATE CURDIR, datadir, msel
     CURDIR = FULLPATH(CURDIR())
    IF !EMPTY(GETENV("CAMD"))
        datadir = FULLPATH(GETENV("CAMD"))
    ELSE
        datadir = CURDIR()
    ENDIF
     desdir = pathstring() + mfname
     SET DEFA TO (datadir)
    IF !EMPTY(GETENV("CAMDUTIL"))
        pkzipcom = "!" + GETENV("CAMDUTIL") + "\PKUNZIP -O &desdir *.db
=> f *.cdx *.idx *.fpt"
    ELSE
        pkzipcom = "!PKUNZIP -O &desdir *.dbf *.cdx *.idx *.fpt"
    ENDIF
     &pkzipcom
     USE
     DELETE FILE t0000000.txt
     SET DEFA TO (CURDIR)

<----RETURN
  CASE  WINDOWS
     ********************

     PRIVATE CURDIR, datadir, msel
     CURDIR = FULLPATH(CURDIR())
    IF !EMPTY(GETENV("KBDATA"))
        datadir = FULLPATH(GETENV("KBDATA"))
    ELSE
        datadir = CURDIR()
    ENDIF
     desdir = pathstring() + mfname
     SET DEFA TO (datadir)
    IF !EMPTY(GETENV("CAMDUTIL"))
        pkzipcom = "!" + GETENV("CAMDUTIL") + "\PKUNZIP -O &desdir *.db
=> f *.cdx *.idx *.fpt"
    ELSE
        pkzipcom = "!PKUNZIP -O &desdir *.dbf *.cdx *.idx *.fpt"
    ENDIF
     &pkzipcom
     USE
     DELETE FILE t0000000.txt
     SET DEFA TO (CURDIR)

<----RETURN
  ENDCASE
*:  EOF: RESTORE.ac1
```

```
*
*
*     08/09/94              DD.SPR              09:39:55
*
*
*     Author's Name
*
*     Copyright (c) 1994 Company Name
*     Address
*     City,     Zip
*
*     Description:
*     This program was automatically generated by GENSCRN.
*
*


#REGION 0
REGIONAL m.currarea, m.talkstat, m.compstat

IF SET("TALK") = "ON"
   SET TALK OFF
   m.talkstat = "ON"
ELSE
   m.talkstat = "OFF"
ENDIF
m.compstat = SET("COMPATIBLE")
SET COMPATIBLE FOXPLUS

m.rborder = SET("READBORDER")
SET readborder ON

*
*
*              Windows Window definitions
*
*
*

IF NOT WEXIST("w_dd") ;
      OR UPPER(WTITLE("W_DD")) == "W_DD.PJX" ;
      OR UPPER(WTITLE("W_DD")) == "W_DD.SCX" ;
      OR UPPER(WTITLE("W_DD")) == "W_DD.MNX" ;
      OR UPPER(WTITLE("W_DD")) == "W_DD.PRG" ;
      OR UPPER(WTITLE("W_DD")) == "W_DD.FRX" ;
      OR UPPER(WTITLE("W_DD")) == "W_DD.QPR"
   DEFINE WINDOW w_dd ;
      AT  0.000, 0.000  ;
      SIZE 30.000,99.200 ;
      TITLE "Question - Data Dictionary" ;
      FONT "MS Sans Serif", 8 ;
      FLOAT ;
      NOCLOSE ;
      MINIMIZE ;
      SYSTEM
   MOVE WINDOW w_dd CENTER
ENDIF


*
*
*          DD/Windows Setup Code - SECTION 2
*
*
*

#REGION 1
SELECT 0
USE area INDEX area
SELECT 0
USE enum INDEX enum
SELECT 0
USE VAL INDEX VAL
SELECT 0
USE dict INDEX dictid, dictname
SET RELATION TO id INTO enum, id INTO VAL
SELECT 0
USE goals INDEX goals
SELECT 0
USE quals INDEX quals
SELECT dict
SET ORDER TO 2
COPY TO ARRAY mdict FIELDS name, id
mq = 1


*
*
*              DD/Windows Screen Layout
*
*
*

#REGION 1
IF WVISIBLE("w_dd")
   ACTIVATE WINDOW w_dd SAME
ELSE
   ACTIVATE WINDOW w_dd NOSHOW
ENDIF
@ 28.077,25.600 GET mbuttons ;
   PICTURE "@*HN \<Edit;\<Add;\<Delete;\<Quit" ;
   SIZE 1.917,7.500,0.750 ;
   DEFAULT 1 ;
   FONT "Terminal", 8 ;
   VALID _qsf0kpt2i()
@ 1.077,7.400 GET mq ;
   PICTURE "@&N" ;
   FROM mdict ;
   SIZE 26.833,50.625 ;
   DEFAULT 1 ;
   FONT "Terminal", 8 ;
   VALID _qsf0kpt7d()

IF NOT WVISIBLE("w_dd")
   ACTIVATE WINDOW w_dd
ENDIF

READ CYCLE MODAL

RELEASE WINDOW w_dd

#REGION 0

SET readborder &rborder

IF m.talkstat = "ON"
   SET TALK ON
ENDIF
IF m.compstat = "ON"
   SET COMPATIBLE ON
ENDIF
```

```

*
*
*             DD/Windows Cleanup Code
*
*
* .

#REGION 1
SELECT area
USE
SELECT enum
USE
SELECT VAL
USE
SELECT dict
USE
SELECT goals
USE
SELECT quals
USE
RETURN

*-----------------------------------
*
* This procedure validate the object to have access to delete or not.
*
*-----------------------------------


*
*
*      DD/Windows Supporting Procedures and Functions
*
*
*

#REGION 1
FUNCTION validobj
PRIVATE msel,mvalid
mvalid = .F.
msel = SELECT()
SELECT quals
SEEK mdict[mq,2]
IF FOUND()
   IF EMPTY(rules) AND EMPTY(ruleso)
      mvalid = .T.
   ELSE
      =errmsg("Delete was not allow, the object was occupied",2)
   ENDIF
ELSE
   mvalid = .T.
ENDIF
SELECT (msel)
RETURN mvalid

FUNCTION qualdel
PRIVATE msel
msel = SELECT()
SELECT dict
SET ORDER TO 1
SEEK mdict[mq,2]
IF FOUND()
   = popupshow("deleting...")
   DO CASE
   CASE dict.datatype $ "ELM"
      SELECT enum
      DELETE FOR id = dict.id
      PACK
   CASE dict.datatype $ "N"
      SELECT VAL
      DELETE FOR id = dict.id
      PACK
   ENDCASE
   SELECT dict
   DELETE
   PACK
   = popuphide()
ENDIF
SELECT (msel)
RETURN

FUNCTION resetdata
PRIVATE msel
msel = SELECT()
SELECT dict
SET ORDER TO 2
COPY TO ARRAY mdict FIELDS name, id
SELECT (msel)
RETURN

*
*
*     _QSFOKPT2I           mbuttons VALID
*
*     Function Origin:
*
*     From Platform:       Windows
*     From Screen:         DD,      Record Number:    2
*     Variable:            mbuttons
*     Called By:           VALID Clause
*     Object Type:         Push Button
*     Snippet Number:      1
*
*
*
FUNCTION _qsf0kpt2i     &&  mbuttons VALID
#REGION 1
DO CASE
CASE mbuttons = 1
   IF mq > 0
      DO ddedit.spr WITH mdict[mq,2], mdict[mq,1]
   ENDIF
CASE mbuttons = 2   && add
   m.adding = .T.
   SELECT dict
   SET ORDER TO 1
   GOTO BOTTOM
   m.id = id + 1
   m.name = ""
   DO ddedit.spr WITH m.id, m.name
CASE mbuttons = 3   && delete
   mdel = validobj()
   IF mdel
      = qualdel()
   ENDIF
CASE mbuttons = 4   && Quit
   mok = .T.
   CLEAR READ
ENDCASE
DO resetdata
```

```
    SHOW GETS

    *
    *
    *       _QSFOKPT7D          mq VALID
    *
    *       Function Origin:
    *
    *       From Platform:      Windows
    *       From Screen:        DD,     Record Number:    3
    *       Variable:           mq
    *       Called By:          VALID Clause
    *       Object Type:        List
    *       Snippet Number:     2
    *
    *
    *
    FUNCTION _qsf0kpt7d     &&  mq VALID
    #REGION 1
  ┌─IF mq > 0
  │    DO ddedit.spr WITH mdict[mq,2], mdict[mq,1]
  └─ENDIF
*:  EOF: DD.ac1
```

```
*
*
*      08/09/94            PLAN.SPR                 09:39:58
*
*
*      Author's Name
*
*      Copyright (c) 1994 Company Name
*      Address
*      City,     Zip
*
*      Description:
*      This program was automatically generated by GENSCRN.
*
*

DO CASE
CASE _WINDOWS


    #REGION 0
    REGIONAL m.currarea, m.talkstat, m.compstat

    IF SET("TALK") = "ON"
       SET TALK OFF
       m.talkstat = "ON"
    ELSE
       m.talkstat = "OFF"
    ENDIF
    m.compstat = SET("COMPATIBLE")
    SET COMPATIBLE FOXPLUS

    m.rborder = SET("READBORDER")
    SET readborder ON

    *
    *
    *              Windows Window definitions
    *
    *
    *

    IF NOT WEXIST("w_goal") ;
          OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.PJX" ;
          OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.SCX" ;
          OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.MNX" ;
          OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.PRG" ;
          OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.FRX" ;
          OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.QPR"
       DEFINE WINDOW w_goal ;
          AT  0.000, 0.000  ;
          SIZE 19.000,64.000 ;
          TITLE "Goal Object Editor" ;
          FONT "MS Sans Serif", 8 ;
          STYLE "B" ;
          FLOAT ;
          CLOSE ;
          SHADOW ;
          NOMINIMIZE
       MOVE WINDOW w_goal CENTER
    ENDIF


    *
    *
    *              PLAN/Windows Screen Layout
    *
    *
    *

    #REGION 1
    IF WVISIBLE("w_goal")
       ACTIVATE WINDOW w_goal SAME
    ELSE
       ACTIVATE WINDOW w_goal NOSHOW
    ENDIF
    @ 16.769,17.000 GET mbuttons ;
       PICTURE "@*HN \<Add;\<OK;\<Cancel" ;
       SIZE 1.769,8.000,1.000 ;
       DEFAULT 1 ;
       FONT "MS Sans Serif", 8 ;
       STYLE "B" ;
       VALID _qsf0kpv8r()
    @ 0.462,2.500 GET mg ;
       PICTURE "@&N" ;
       FROM mgoals ;
       SIZE 15.000,71.000 ;
       DEFAULT 1 ;
       FONT "MS Sans Serif", 8 ;
       VALID _qsf0kpvcr()

    IF NOT WVISIBLE("w_goal")
       ACTIVATE WINDOW w_goal
    ENDIF

    READ CYCLE MODAL

    RELEASE WINDOW w_goal

    #REGION 0

    SET readborder &rborder

    IF m.talkstat = "ON"
       SET TALK ON
    ENDIF
    IF m.compstat = "ON"
       SET COMPATIBLE ON
    ENDIF


CASE _DOS


    #REGION 0
    REGIONAL m.currarea, m.talkstat, m.compstat

    IF SET("TALK") = "ON"
       SET TALK OFF
       m.talkstat = "ON"
    ELSE
```

```
      m.talkstat = "OFF"
   ENDIF
   m.compstat = SET("COMPATIBLE")
   SET COMPATIBLE FOXPLUS

   *
   *
   *              MS-DOS Window definitions
   *
   *
   *

   IF NOT WEXIST("w_goal") ;
         OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.PJX" ;
         OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.SCX" ;
         OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.MNX" ;
         OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.PRG" ;
         OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.FRX" ;
         OR UPPER(WTITLE("W_GOAL")) == "W_GOAL.QPR"
      DEFINE WINDOW w_goal ;
         FROM INT((SROW()-19)/2),INT((SCOL()-64)/2) ;
         TO INT((SROW()-19)/2)+18,INT((SCOL()-64)/2)+63 ;
         TITLE "Goal Object Editor" ;
         FLOAT ;
         CLOSE ;
         SHADOW ;
         NOMINIMIZE ;
         COLOR SCHEME 1
   ENDIF

   *
   *
   *              PLAN/MS-DOS Screen Layout
   *
   *
   *

   #REGION 1
   IF WVISIBLE("w_goal")
      ACTIVATE WINDOW w_goal SAME
   ELSE
      ACTIVATE WINDOW w_goal NOSHOW
   ENDIF
   @ 16,17 GET mbuttons ;
      PICTURE "@*HN \<Add;\<OK;\<Cancel" ;
      SIZE 1,8,1 ;
      DEFAULT 1 ;
      VALID _qsf0kpvqb()
   @ 1,1 GET mg ;
      PICTURE "@&N" ;
      FROM mgoals ;
      SIZE 14,60 ;
      DEFAULT 1 ;
      VALID _qsf0kpvw8() ;
      COLOR SCHEME 2

   IF NOT WVISIBLE("w_goal")
      ACTIVATE WINDOW w_goal
   ENDIF

   READ CYCLE MODAL

   RELEASE WINDOW w_goal

   #REGION 0
   IF m.talkstat = "ON"
      SET TALK ON
   ENDIF
   IF m.compstat = "ON"
      SET COMPATIBLE ON
   ENDIF


ENDCASE


*
*
*      _QSFOKPV8R          mbuttons VALID
*
*      Function Origin:
*
*      From Platform:      Windows
*      From Screen:        PLAN,     Record Number:    2
*      Variable:           mbuttons
*      Called By:          VALID Clause
*      Object Type:        Push Button
*      Snippet Number:     1
*
*
*
FUNCTION _qsf0kpv8r     &&  mbuttons VALID
#REGION 1
DO CASE
CASE mbuttons = 1
   DEACTIVATE WINDOW w_goal
   DO object.spr WITH "G"
   ACTIVATE WINDOW w_goal
   DO setgoals
CASE mbuttons = 2   && ok
   mok = .T.
   CLEAR READ
CASE mbuttons = 3   && cancel
   mok = .F.
   CLEAR READ
ENDCASE
SHOW GETS


*
*
*      _QSFOKPVCR          mg VALID
*
*      Function Origin:
*
*      From Platform:      Windows
*      From Screen:        PLAN,     Record Number:    3
*      Variable:           mg
*      Called By:          VALID Clause
*      Object Type:        List
*      Snippet Number:     2
```

```
*
*
*
FUNCTION _qsf0kpvcr     &&  mg VALID
#REGION 1
DO edgoal

*
*
*       _QSFOKPVQB           mbuttons VALID
*
*       Function Origin:
*
*       From Platform:       MS-DOS
*       From Screen:         PLAN,     Record Number:    7
*       Variable:            mbuttons
*       Called By:           VALID Clause
*       Object Type:         Push Button
*       Snippet Number:      3
*
*
*
FUNCTION _qsf0kpvqb     &&  mbuttons VALID
#REGION 1
DO CASE
CASE mbuttons = 1
   DEACTIVATE WINDOW w_goal
   DO object.spr WITH "G"
   ACTIVATE WINDOW w_goal
   DO setgoals
CASE mbuttons = 2   && ok
   mok = .T.
   CLEAR READ
CASE mbuttons = 3   && cancel
   mok = .F.
   CLEAR READ
ENDCASE
SHOW GETS


*
*
*       _QSFOKPVW8           mg VALID
*
*       Function Origin:
*
*       From Platform:       MS-DOS
*       From Screen:         PLAN,     Record Number:    8
*       Variable:            mg
*       Called By:           VALID Clause
*       Object Type:         List
*       Snippet Number:      4
*
*
*
FUNCTION _qsf0kpvw8     &&  mg VALID
#REGION 1
DO edgoal
*:  EOF: PLAN.ac1
```

```
*:*************************************************************************
=> *******
*:
*: Procedure file: C:\CAMD2\KBEDIT\WORKW\ERRMSG.PRG
*:         System: Knowledge Base Editor
*:         Author: Hoa L. Ly
*:      Copyright (c) June 1, 1994, Naval Health Research Center, Code 2
=> 2
*:  Last modified: 07/29/94 at  9:22:58
*:
*:         Set by: ACTION.SPR
*:               : ACTELSE.SPR
*:               :  QUIT()             (function in KBINIT.PRG)
*:               : VALIDOBJ()          (function in QUAL.SPR)
*:
*:      Documented 15:01:10                                  FoxDoc versio
=> n 3.00a
*:*************************************************************************
=> *******

    * Open a message window
    *
    * ERRMSG( <expC>[, <expN>] )
    *
    PARAMETERS errmsg, timelimit
    PRIVATE ALL
    * If no errmessage is sent then quit
   ┌IF PARAMETERS()=0
<══════RETURN
   └ENDIF

    * m.TimeLimit is used in a WAIT TIMEOUT command
    * to control the amount of time ERRMSG display's its message
    * If no time limit is received set time to 0 wait forever

   ┌IF PARAMETERS()=1 OR EMPTY(m.timelimit)
   │   m.timelimit=0
   └ENDIF

   ┌IF TYPE("m.TimeLimit")!="N"
   │   m.timelimit=IIF(TYPE("m.TimeLImit")="C",INT(VAL(m.timelimit)),0)
   └ENDIF

    m.errmsg=IIF(TYPE("m.errmsg")!="C","",m.errmsg)

   ┌IF PARAMETERS()=2 AND EMPTY(m.errmsg) AND m.timelimit>0
<══════RETURN
   └ENDIF

    * set talk off
   ┌IF SET('TALK') = 'ON'  && TALK handled as a special case.
   │   SET TALK OFF          && Turn TALK OFF
   │   savetalk = 'ON'       && TALK was ON, save the setting
   ├ELSE   && TALK is OFF
   │   savetalk = 'OFF'      && TALK was OFF, save the setting
   └ENDIF

    * Get the length of the message to size window
    m.len=LEN(m.errmsg)
    m.high = 1
    m.old = SET("memowidth")
    SET MEMOWIDTH TO 75
    * Set miminum length for the Press any key
   ┌IF m.timelimit = 0
   │   m.len=IIF(m.len<30,30,m.len)
   └ENDIF

   ┌IF m.len >75
   │   m.high=MEMLINES(m.errmsg)
   │   m.len=75
   └ENDIF

    *Find begining/ending of the window
    m.begin=40-(INT(m.len/2)+1)
    m.end=40+(INT(m.len/2)+1)

    * Remember the current window status
    m.oldwindow =IIF( WOUTPUT() = errmsg, "" , WOUTPUT() )

    * Create the window
    DEFINE WINDOW errmsg ;
       AT 22, BEGIN;
       SIZE m.high + 3, m.len + 5 ;
       FONT "Terminal", 8 ;
       FLOAT ;
       NOCLOSE ;
       MINIMIZE ;
       DOUBLE ;
       COLOR RGB(,,,0,0,255)

    *define window errmsg from 18,m.begin to 23,m.end color scheme 1
    ACTIVATE WINDOW errmsg

    * Print the message centered in the window

    CLEAR
    @ 1, ( WCOL() - m.len )/2 SAY m.errmsg ;
       SIZE (m.high), (m.len);
       FONT "Terminal",8

   ┌IF m.timelimit = 0
   │   *   m.presskey = "Press any key to continue"
   │   *   @ 2, ( wcol() - len( m.presskey ) )/2 say m.presskey
   │  ┌IF m.high > 1
   │  │   @ ROW(), 1 SAY ""
   │  ├ELSE
   │  │   @ ROW() + 1, 1 SAY ""
   │  └ENDIF
   │   WAIT
   ├ELSE
   │   *   Wait for the number of seconds in m.Timeout
   │   *   A value of 0 will wait forever
   │   WAIT "" TIMEOUT m.timelimit
   └ENDIF


    * Close Window
    RELEASE WINDOW errmsg

    * If there was no output window originally
   ┌IF EMPTY( m.oldwindow )
   │
   │   * Send future output back to the screen
   │   ACTIVATE SCREEN
   │
   ├ELSE
   │
   │   * Return output to the original window
   │   ACTIVATE WINDOW ( m.oldwindow )
   └ENDIF

    SET TALK &savetalk   && Restore original TALK setting
```

```
<——RETURN
 *:  EOF: ERRMSG.act
```

```
*
*    08/09/94              DICT.SPR                 09:40:02
*
*
*
*    Author's Name
*
*    Copyright (c) 1994 Company Name
*    Address
*    City,     Zip
*
*    Description:
*    This program was automatically generated by GENSCRN.
*
*

PARAMETERS m.newid

*
*
*              DICT/Windows Setup Code - SECTION 1
*
*
*

#REGION 1
PRIVATE mp, mpn, msel, m.adding
DIMENSION enum[20], enumr[20]
msel = SELECT()
SELECT dict
enum = " "
enumr = 0
mp = 1
m.adding = .F.
m.qrecno = RECNO()
SEEK m.newid
IF !FOUND()
   m.adding = .T.
   SCATTER MEMVAR BLANK
   m.id = m.newid
   SHOW GET m.datatype ENABLE
ELSE
   SCATTER MEMVAR
ENDIF
DO setenum
*do setval

#REGION 0
REGIONAL m.currarea, m.talkstat, m.compstat

IF SET("TALK") = "ON"
   SET TALK OFF
   m.talkstat = "ON"
ELSE
   m.talkstat = "OFF"
ENDIF
m.compstat = SET("COMPATIBLE")
SET COMPATIBLE FOXPLUS

m.rborder = SET("READBORDER")
SET readborder ON

*
*
*                Windows Window definitions
*
*
*

IF NOT WEXIST("dict") ;
      OR UPPER(WTITLE("DICT")) == "DICT.PJX" ;
      OR UPPER(WTITLE("DICT")) == "DICT.SCX" ;
      OR UPPER(WTITLE("DICT")) == "DICT.MNX" ;
      OR UPPER(WTITLE("DICT")) == "DICT.PRG" ;
      OR UPPER(WTITLE("DICT")) == "DICT.FRX" ;
      OR UPPER(WTITLE("DICT")) == "DICT.QPR"
   DEFINE WINDOW dict ;
      AT  0.000, 0.000  ;
      SIZE 26.250,68.875 ;
      TITLE "Dictionary Editor" ;
      FONT "Terminal", 8 ;
      FLOAT ;
      NOCLOSE ;
      MINIMIZE ;
      SYSTEM ;
      COLOR RGB(,,,0,255,255)
   MOVE WINDOW dict CENTER
ENDIF


*
*
*              DICT/Windows Screen Layout
*
*
*

#REGION 1
IF WVISIBLE("dict")
   ACTIVATE WINDOW dict SAME
ELSE
   ACTIVATE WINDOW dict NOSHOW
ENDIF
@ 1.333,55.250 SAY "Type"  ;
   FONT "Terminal", 8
@ 22.167,10.500 SAY "Hi"  ;
   FONT "Terminal", 8
@ 22.167,30.000 SAY "Lo"  ;
   FONT "Terminal", 8
@ 22.167,48.000 SAY "Units"  ;
   FONT "Terminal", 8
@ 1.333,3.000 SAY "id"  ;
   FONT "Terminal", 8
@ 1.417,12.000 SAY "Name"  ;
   FONT "Terminal", 8
@ 20.167,9.000 SAY "Width"  ;
   FONT "Terminal", 8
@ 20.167,30.000 SAY "Decimals"  ;
   FONT "Terminal", 8
@ 22.083,3.000 SAY "Range:"  ;
   FONT "Terminal", 8
@ 8.667,3.375 SAY "Enum"  ;
   FONT "Terminal", 8
@ 20.167,3.000 SAY "Val:"  ;
   FONT "Terminal", 8
@ 1.167,6.250 GET m.id ;
   SIZE 1.000,3.125 ;
   DEFAULT " " ;
   FONT "Terminal", 8 ;
   DISABLE
@ 1.167,18.250 GET m.name ;
```

```
        SIZE 1.000,33.875 ;
        DEFAULT " " ;
        FONT "Terminal", 8
     @ 1.083,60.500 GET m.datatype ;
        PICTURE "@^ N;E;M;L" ;
        SIZE 1.500,4.375 ;
        DEFAULT "N" ;
        FONT "Terminal", 8 ;
        VALID _qsf0kpy4x() ;
        DISABLE
     @ 3.167,21.250 GET dict.askable ;
        SIZE 1.000,4.625 ;
        DEFAULT .F. ;
        FONT "Terminal", 8
     @ 5.000,3.250 EDIT m.question ;
        SIZE 3.000,61.750,0.000 ;
        DEFAULT " " ;
        FONT "Terminal", 8 ;
        SCROLL ;
        WHEN dict.askable
     @ 10.167,3.125 GET mp ;
        PICTURE "@&N" ;
        FROM enum ;
        SIZE 9.333,62.000 ;
        DEFAULT 1 ;
        FONT "Terminal", 8 ;
        WHEN m.datatype $ "EML" ;
        VALID _qsf0kpy8w()
     @ 20.167,16.750 GET m.width ;
        SIZE 1.000,5.500 ;
        DEFAULT 0 ;
        FONT "Terminal", 8 ;
        WHEN m.datatype = "N" ;
        DISABLE
     @ 20.167,39.250 GET m.dec ;
        SIZE 1.000,11.500 ;
        DEFAULT 0 ;
        FONT "Terminal", 8 ;
        WHEN m.datatype = "N" ;
        DISABLE
     @ 22.167,16.750 GET m.hi ;
        SIZE 1.000,10.000 ;
        DEFAULT 0 ;
        FONT "Terminal", 8 ;
        WHEN m.datatype = "N" ;
        DISABLE
     @ 22.167,33.250 GET m.lo ;
        SIZE 1.000,13.000 ;
        DEFAULT 0 ;
        FONT "Terminal", 8 ;
        WHEN m.datatype = "N" ;
        DISABLE
     @ 22.167,54.250 GET m.units ;
        SIZE 1.000,10.000 ;
        DEFAULT " " ;
        FONT "Terminal", 8 ;
        WHEN m.datatype = "N" ;
        DISABLE
     @ 24.000,25.500 GET mbuttons ;
        PICTURE "@*HT OK;Cancel" ;
        SIZE 1.917,8.375,0.750 ;
        DEFAULT 1 ;
        FONT "Terminal", 8 ;
        VALID _qsf0kpyf6()
     @ 3.167,3.000 SAY "Question Askable:"  ;
        FONT "Terminal", 8 ;
```

```
        STYLE "T"

     IF NOT WVISIBLE("dict")
        ACTIVATE WINDOW dict
     ENDIF

     READ CYCLE MODAL ;
        WHEN _qsf0kpyj0()

     RELEASE WINDOW dict

     #REGION 0

     SET readborder &rborder

     IF m.talkstat = "ON"
        SET TALK ON
     ENDIF
     IF m.compstat = "ON"
        SET COMPATIBLE ON
     ENDIF


     *
     *
     *               DICT/Windows Cleanup Code
     *
     *
     *

     #REGION 1
     SELECT dict
     GOTO m.qrecno
  <──RETURN



     *
     *
     *        DICT/Windows Supporting Procedures and Functions
     *
     *
     *

     #REGION 1
     PROCEDURE edenum
     SELECT enum
     IF enumr[mp] > 0
        GOTO enumr[mp]
     ENDIF
     *m.id = dict.id
     DO enum.spr
     DO setenum
     SHOW GETS
     SELECT dict
  <──RETURN


     PROCEDURE setenum
     enum = " "
     enumr = 0
     IF m.datatype $ "EML"
        PRIVATE msel, i
        msel = SELECT()
        SELECT enum
        SEEK m.id
```

```
    i = 0
   DO WHILE id = m.id
      i = i + 1
      enumr[i] = RECNO()
      enum[i] = mutex + " " + TRIM( enumerate )
      SKIP
   ENDDO
   mpn = i
   mp = MIN(mpn, mp)
   mp = MAX(1, mp)
   IF enumr[mp] > 0
      GOTO enumr[mp]
   ENDIF
   SELECT (msel)
ENDIF
RETURN

PROCEDURE setval
IF m.datatype $ "N"
   PRIVATE msel, i
   msel = SELECT()
   SELECT VAL
   SCATTER MEMVAR
   SHOW GETS
   SELECT (msel)
ENDIF
RETURN


*
*
*        _QSFOKPY4X            m.datatype VALID
*
*        Function Origin:
*
*        From Platform:        Windows
*        From Screen:          DICT,      Record Number:   15
*        Variable:             m.datatype
*        Called By:            VALID Clause
*        Object Type:          Popup
*        Snippet Number:       1
*
*
*
FUNCTION _qsf0kpy4x     &&  m.datatype VALID
#REGION 1
SHOW GET m.datatype DISABLE

*
*
*        _QSFOKPY8W            mp VALID
*
*        Function Origin:
*
*        From Platform:        Windows
*        From Screen:          DICT,      Record Number:   18
*        Variable:             mp
*        Called By:            VALID Clause
*        Object Type:          List
*        Snippet Number:       2
*
*
*
FUNCTION _qsf0kpy8w     &&  mp VALID
#REGION 1
DO edenum

*
*
*        _QSFOKPYF6            mbuttons VALID
*
*        Function Origin:
*
*        From Platform:        Windows
*        From Screen:          DICT,      Record Number:   24
*        Variable:             mbuttons
*        Called By:            VALID Clause
*        Object Type:          Push Button
*        Snippet Number:       3
*
*
*
FUNCTION _qsf0kpyf6     &&  mbuttons VALID
#REGION 1
DO CASE
CASE mbuttons = 1   && ok
   SELECT dict
   GATHER MEMVAR
   IF m.datatype $ "N"
      SELECT VAL
      GATHER MEMVAR
   ENDIF
CASE mbuttons = 2   && cancel
   mok = .F.
ENDCASE

*
*
*        _QSFOKPYJ0            Read Level When
*
*        Function Origin:
*
*
*        From Platform:        Windows
*        From Screen:          DICT
*        Called By:            READ Statement
*        Snippet Number:       4
*
*
*
*
FUNCTION _qsf0kpyj0     && Read Level When
*
* When Code from screen: DICT
*
#REGION 1
IF m.datatype $ "N"
   PRIVATE msel
   msel = SELECT()
   SELECT VAL
   SCATTER MEMVAR
   SHOW GET mp DISABLE
   SHOW GET m.width ENABLE
   SHOW GET m.dec ENABLE
   SHOW GET m.hi ENABLE
   SHOW GET m.lo ENABLE
   SHOW GET m.units ENABLE
   SHOW GETS
ENDIF
*:  EOF: DICT.ac1
```

```
*
*
*        08/09/94              KBEDIT.SPR               09:40:05
*
*
*        Author's Name
*
*        Copyright (c) 1994 Company Name
*        Address
*        City,     Zip
*
*        Description:
*        This program was automatically generated by GENSCRN.
*
*


#REGION 0
REGIONAL m.currarea, m.talkstat, m.compstat

IF SET("TALK") = "ON"
   SET TALK OFF
   m.talkstat = "ON"
ELSE
   m.talkstat = "OFF"
ENDIF
m.compstat = SET("COMPATIBLE")
SET COMPATIBLE FOXPLUS

m.rborder = SET("READBORDER")
SET readborder ON

*
*
*                    Windows Window definitions
*
*
*

IF NOT WEXIST("kbedit") ;
      OR UPPER(WTITLE("KBEDIT")) == "KBEDIT.PJX" ;
      OR UPPER(WTITLE("KBEDIT")) == "KBEDIT.SCX" ;
      OR UPPER(WTITLE("KBEDIT")) == "KBEDIT.MNX" ;
      OR UPPER(WTITLE("KBEDIT")) == "KBEDIT.PRG" ;
      OR UPPER(WTITLE("KBEDIT")) == "KBEDIT.FRX" ;
      OR UPPER(WTITLE("KBEDIT")) == "KBEDIT.QPR"
   DEFINE WINDOW kbedit ;
      AT  0.000, 0.000 ;
      SIZE 29.667,64.250 ;
      TITLE "Knowledge Base Rule Editor" ;
      FONT "Terminal", 8 ;
      FLOAT ;
      NOCLOSE ;
      MINIMIZE ;
      SYSTEM ;
      COLOR RGB(,,,128,128,128)
   MOVE WINDOW kbedit CENTER
ENDIF


*
*
*          KBEDIT/Windows Setup Code - SECTION 2
*
*
*

#REGION 1
PRIVATE mp, me, ma, mpn, man, ntriples,ptop
DIMENSION prem[50], premr[50], triple[50,5]
DIMENSION premo[50], prems[50], premm[50]
DIMENSION act[50], actr[50], actelse[50], actelser[50]

PRIVATE msel
msel = SELECT()
SELECT 0
USE fact INDEX fact
SELECT 0
USE premise INDEX premise
SET RELATION TO fact INTO fact
SELECT 0
USE action INDEX action
SELECT 0
USE rule INDEX rulearea,salience,rule
SET RELATION TO premise INTO premise, action INTO action
SELECT area
SET RELATION TO area INTO rule
SELECT rule
SET TOPIC TO "RULE"
mp   = 1
act  = " "
actr = 0
ma   = 1
me = 1
DO setprem
DO setact
DO setelse
DO setaction



*
*
*                 KBEDIT/Windows Screen Layout
*
*
*

#REGION 1
IF WVISIBLE("kbedit")
   ACTIVATE WINDOW kbedit SAME
ELSE
   ACTIVATE WINDOW kbedit NOSHOW
ENDIF
@ 3.167,1.625 SAY "Rule:"  ;
   FONT "Terminal", 8 ;
   STYLE "T"
@ 1.167,18.250 GET m.areaname ;
   SIZE 1.000,34.000 ;
   DEFAULT " " ;
   FONT "Terminal", 8 ;
   DISABLE
@ 3.167,7.750 GET rule.rule ;
   SIZE 1.000,25.000 ;
   DEFAULT 0 ;
   FONT "Terminal", 8 ;
   DISABLE
@ 5.000,1.500 GET minvprem ;
   PICTURE "@*HN \<IF:" ;
   SIZE 1.500,7.250,0.750 ;
   DEFAULT 1 ;
```

```
        FONT "Terminal", 8 ;
        VALID _qsf0kq0sa() ;
        MESSAGE "Premises"
     @ 7.167,1.625 GET mp ;
        PICTURE "@&N" ;
        FROM prem ;
        SIZE 4.667,50.875 ;
        DEFAULT 1 ;
        FONT "Terminal", 8
     @ 13.000,1.500 GET minvact ;
        PICTURE "@*HN \<THEN:" ;
        SIZE 1.583,7.125,0.750 ;
        DEFAULT 1 ;
        FONT "Terminal", 8 ;
        VALID _qsf0kq0wk() ;
        MESSAGE "Actions"
     @ 15.167,1.625 GET ma ;
        PICTURE "@&N" ;
        FROM act ;
        SIZE 4.667,50.875 ;
        DEFAULT 1 ;
        FONT "Terminal", 8
     @ 21.000,1.500 GET minvelse ;
        PICTURE "@*HN \<ELSE:" ;
        SIZE 1.583,7.250,0.750 ;
        DEFAULT 1 ;
        FONT "Terminal", 8 ;
        VALID _qsf0kq10r() ;
        MESSAGE "Actions (Else)"
     @ 23.167,1.625 GET me ;
        PICTURE "@&N" ;
        FROM actelse ;
        SIZE 4.667,50.875 ;
        DEFAULT 1 ;
        FONT "Terminal", 8
     @ 1.000,54.000 GET mbbutton ;
        PICTURE "@*VN \<Next;\<Prev;\<Browse;\<Edit;e\<Xit" ;
        SIZE 1.750,8.875,1.083 ;
        DEFAULT 1 ;
        FONT "Terminal", 8 ;
        VALID _qsf0kq14z()
     @ 1.167,1.500 SAY "Knoledge Base:"  ;
        FONT "Terminal", 8 ;
        STYLE "T"

    ┌IF NOT WVISIBLE("kbedit")
    │  ACTIVATE WINDOW kbedit
    └ENDIF

     READ CYCLE ;
        WHEN setaction()

     RELEASE WINDOW kbedit

     #REGION 0

     SET readborder &rborder

    ┌IF m.talkstat = "ON"
    │   SET TALK ON
    └ENDIF
    ┌IF m.compstat = "ON"
    │   SET COMPATIBLE ON
    └ENDIF

```

```
     *
     *
     *          KBEDIT/Windows Cleanup Code
     *
     *
     *

     #REGION 1
     SELECT rule
     USE
     SELECT premise
     USE
     SELECT fact
     USE
     SELECT action
     USE
     SELECT (msel)
  <──RETURN



     *
     *
     *          KBEDIT/Windows Supporting Procedures and Functions
     *
     *
     *

     #REGION 1
     PROCEDURE setprem
     PRIVATE msel, i, j, jj, K, N, p, R, s
     msel = SELECT()
     SELECT premise
     SEEK rule.premise
     i = 0
     N = 0
     prem = " "

     * collect all triples
    ┌DO WHILE clause = rule.premise .AND. !EOF()
    │   i = i + 1
    │   triple[i,1] = premise.op
    │   triple[i,2] = premise.fact
    │   triple[i,3] = premise.factr
    │   triple[i,4] = 0
    │   triple[i,5] = RECNO()
    │   SKIP
    └ENDDO
     ntriples = i

     * collect leaf nodes
     SELECT fact
    ┌FOR j = 1 TO ntriples
    │ ┌IF EMPTY( triple[j,1] )
    │ │   SEEK triple[j,2]
    │ │ ┌IF FOUND()
    │ │ │   N = N + 1
    │ │ │ ┌IF OBJECT = "S"
    │ │ │ │   SELECT dict
    │ │ │ ├ELSE
    │ │ │ │   SELECT disease
    │ │ │ └ENDIF
    │ │ │   SEEK fact.id
    │ │ │   m.name = name
    │ │ │   SELECT fact
    │ │ │   m.val = TRIM( IIF( TAG = "T", LEFT( TEXT, 80), VAL ) )
```

```
            SELECT dict

          FOR jj = 1 TO 2
             K = AT ( "@", m.val, jj )
            IF K = 0
               EXIT
            ENDIF
             p = VAL( SUBSTR( m.val, K + 1 ) )
             SEEK p
             m.val = STRTRAN( m.val, "@" + LTRIM( STR( p ) ), "(" + TR
=> IM( name ) + ")"   )
          NEXT

          SELECT fact
          prem[n] = TRIM( m.name ) + " " + op + " " + m.val
          premr[n] = j
          premo[n] = ""
          triple[j,4] = N
        ENDIF
      ENDIF
    NEXT
    SELECT premise

    premm = .F.
    ptop = 0
    IF ntriples > 0
       =PUSH(ntriples)      && root of expression tree
    ENDIF

    * parse the expression tree
    DO WHILE !sempty()
       i = POP()
      IF !EMPTY( triple[i, 1] )
         * operator node
        IF !premm[i]
           * not yet marked; defer
           premm[i] = .T.
           =PUSH( i )                  && push operator
           =PUSH( triple[i, 3] )       && ..right
           =PUSH( triple[i, 2] )       && ..left
        ELSE
           * already marked..
           j = triple[triple[i, 2], 4]  && left term
           K = triple[triple[i, 3], 4]  && right term
           triple[i, 4] = j             && right term
          IF LEFT( prem[k], 1 ) = triple[i,1]
              * same operator; fold tree
              s = hlink( SUBSTR( prem[j], 3 ) )  && add link
              prem[j] = "┬─" + s                && add elbow
              premo[k] = IIF(EMPTY(premo[k]),ALLTRIM(STR(i)),premo[k] +
=>  "^" + ALLTRIM(STR(i)))
          ELSE
              s = hlink( prem[j] )
              prem[j] = "┌─" + s         && complete left child
              s = hlink(prem[k] )
              prem[k] = triple[i,1] + "─" + s  && opcode and right chil
=> d
              premo[k] = IIF(EMPTY(premo[k]),ALLTRIM(STR(i)),premo[k] +
=>  "^" + ALLTRIM(STR(i)))
              * indent remaining terms
            FOR R = 1 TO N
              IF R <> j  .AND. R <> K
                 * indent
                 prem[r] = SPACE(2) + prem[r]
              ENDIF
            NEXT
          ENDIF
           * bridge missing links
          FOR R = 1 TO N
            IF R > j .AND. R < K
               s = prem[r]
              IF EMPTY( LEFT( s, 1 ) )
                 prem[r] = "│" + SUBSTR( s, 2 )
              ENDIF
            ENDIF
          NEXT
           * cleanup
          FOR R = j TO K-1
             * complete missing links
             =vlink( "┌", "┌" )
             =vlink( "├", "┌" )
             =vlink( "│", "┌" )
          NEXT
        ENDIF
      ENDIF
    ENDDO

    mpn = N
    mp  = MIN( mpn, mp )
    mp  = MAX( 1, mp )
    SELECT (msel)
 <──RETURN

    FUNCTION vlink
    PARAMETERS FROM, TO
    PRIVATE i,s
    i = 1
    DO WHILE EMPTY( SUBSTR( prem[r], i, 1) )
       i = i + 1                 && find leading non-blank
    ENDDO
    s = prem[r+1]
    * complete missing links
    IF SUBSTR( prem[r], i, 1 ) = FROM .AND. SUBSTR( s, i, 1 ) = TO
       prem[r+1] = LEFT( s, i-1 ) + "├" + SUBSTR( s, i+1 )
    ENDIF
 <──RETURN ""

    FUNCTION hlink
    PARAMETERS s
    PRIVATE p
    p = ""
    DO WHILE EMPTY( LEFT(s, 2 ) )
       p = p + "──"
       s = SUBSTR( s, 3 )
    ENDDO
 <──RETURN p + s

    FUNCTION PUSH
    PARAMETERS N
    ptop = ptop + 1
    prems[ptop] = N
 <──RETURN N

    FUNCTION POP
    PRIVATE N
    N = prems[ptop]
    ptop = ptop - 1
 <──RETURN N

    FUNCTION sempty
 <──RETURN ptop < 1
```

```
PROCEDURE setact
PRIVATE msel, i, j, K, p
msel = SELECT()
SELECT action
SEEK rule.action
i = 0
act = " "
actr = 0
DO WHILE clause = rule.action .AND. !EOF()
   i = i + 1
   actr[i] = RECNO()
   IF OBJECT = "D"
      SELECT disease
   ELSE
      SELECT dict
   ENDIF
   SEEK action.id
   m.name = name
   SELECT action
   m.val = TRIM( IIF( TAG = "T", LEFT( TEXT, 80), VAL ) )
   SELECT dict
   FOR j = 1 TO 2
      K = AT ( "@", m.val, j )
      IF K = 0
         EXIT
      ENDIF
      p = VAL( SUBSTR( m.val, K + 1 ) )
      SEEK p
      m.val = STRTRAN( m.val, "@" + LTRIM( STR( p ) ), "(" + TRIM( na
=> me ) + ")"   )
   NEXT
   SELECT action
   act[i] = TRIM( m.name ) + " " + op + " " + m.val
   SKIP
ENDDO
man = i
ma = MIN(man, ma)
ma = MAX(1, ma)
IF actr[ma] > 0
   GOTO actr[ma]
ENDIF
SELECT (msel)
RETURN

PROCEDURE setelse
PRIVATE msel, i, j, K, p
msel = SELECT()
SELECT action
SEEK rule.else
i = 0
actelse = " "
actelser = 0
DO WHILE clause = rule.else .AND. !EOF()
   i = i + 1
   actelser[i] = RECNO()
   IF OBJECT = "D"
      SELECT disease
   ELSE
      SELECT dict
   ENDIF
   SEEK action.id
   m.name = name
   SELECT action
   m.val = TRIM( IIF( TAG = "T", LEFT( TEXT, 80), VAL ) )
   SELECT dict
   FOR j = 1 TO 2
      K = AT ( "@", m.val, j )
      IF K = 0
         EXIT
      ENDIF
      p = VAL( SUBSTR( m.val, K + 1 ) )
      SEEK p
      m.val = STRTRAN( m.val, "@" + LTRIM( STR( p ) ), "(" + TRIM( na
=> me ) + ")"   )
   NEXT
   SELECT action
   actelse[i] = TRIM( m.name ) + " " + op + " " + m.val
   SKIP
ENDDO
man = i
IF EMPTY(man)
   actelse[1] = ""
   actelser[1] = 0
ELSE
   me = MIN(man, me)
   me = MAX(1, me)
   IF actelser[me] > 0
      GOTO actelser[me]
   ENDIF
ENDIF
SELECT (msel)
RETURN

FUNCTION setaction
IF EMPTY(prem[1])
   SHOW GET minvprem DISABLE
   SHOW GET mp DISABLE
ELSE
   SHOW GET minvprem ENABLE
   SHOW GET mp ENABLE
ENDIF
IF EMPTY(act[1])
   SHOW GET minvact DISABLE
   SHOW GET ma DISABLE
ELSE
   SHOW GET minvact ENABLE
   SHOW GET ma ENABLE
ENDIF
IF EMPTY(actelse[1])
   SHOW GET minvelse DISABLE
   SHOW GET me DISABLE
ELSE
   SHOW GET minvelse ENABLE
   SHOW GET me ENABLE
ENDIF
RETURN

   *
   *
   *      _QSFOKQOSA          minvprem VALID
   *
   *      Function Origin:
   *
   *      From Platform:      Windows
   *      From Screen:        KBEDIT,      Record Number:   5
   *      Variable:           minvprem
   *      Called By:          VALID Clause
   *      Object Type:        Push Button
   *      Snippet Number:     1
   *
   *
   *
```

```
FUNCTION _qsf0kq0sa     &&  minvprem VALID
#REGION 1
PRIVATE msel
msel = SELECT()
SELECT premise
LOCATE FOR clause = rule.premise
SET TOPIC TO "PREMISE"
DO term.spr
DO setprem
SELECT (msel)
SHOW GETS

*
*
*      _QSFOKQOWK          minvact VALID
*
*      Function Origin:
*
*      From Platform:      Windows
*      From Screen:        KBEDIT,     Record Number:    7
*      Variable:           minvact
*      Called By:          VALID Clause
*      Object Type:        Push Button
*      Snippet Number:     2
*
*
*
FUNCTION _qsf0kq0wk     &&  minvact VALID
#REGION 1
PRIVATE msel
msel = SELECT()
SELECT action
LOCATE FOR clause = rule.action
SET TOPIC TO "ACTION"
DO action.spr
SELECT (msel)
SHOW GETS

*
*
*      _QSFOKQ10R          minvelse VALID
*
*      Function Origin:
*
*      From Platform:      Windows
*      From Screen:        KBEDIT,     Record Number:    9
*      Variable:           minvelse
*      Called By:          VALID Clause
*      Object Type:        Push Button
*      Snippet Number:     3
*
*
*
FUNCTION _qsf0kq10r     &&  minvelse VALID
#REGION 1
PRIVATE msel
msel = SELECT()
SELECT action
LOCATE FOR clause = rule.else
SET TOPIC TO "ACTION"
DO actelse.spr
DO setelse
SHOW GETS
SELECT (msel)

*
*
*      _QSFOKQ14Z          mbbutton VALID
*
*      Function Origin:
*
*      From Platform:      Windows
*      From Screen:        KBEDIT,     Record Number:    11
*      Variable:           mbbutton
*      Called By:          VALID Clause
*      Object Type:        Push Button
*      Snippet Number:     4
*
*
*
FUNCTION _qsf0kq14z     &&  mbbutton VALID
#REGION 1
SELECT rule
DO CASE
CASE mbbutton = 1            && <Next>
    m.recno = RECNO()
    IF !EOF()
        SKIP
    ELSE
        GOTO BOTTOM
    ENDIF
    m.goback = area != m.areaid
    IF m.goback
        GOTO m.recno
    ENDIF
CASE mbbutton = 2            && <previous>
    m.recno = RECNO()
    IF !BOF()
        SKIP -1
    ELSE
        GOTO TOP
    ENDIF
    m.goback  = area != m.areaid
    IF m.goback
        GOTO m.recno
    ENDIF
CASE mbbutton = 3            && <Browse>
    SET TOPIC TO "BROWSE"
    BROWSE FIELDS rule,salience FOR area = m.areaid NOEDIT
CASE mbbutton = 4            && <Edit>
    SET TOPIC TO "EDIT"
    DO rule.spr
CASE mbbutton = 5            && <Quit>
    CLEAR READ
<---RETURN
ENDCASE
SCATTER MEMVAR
DO setprem
DO setact
DO setelse
DO setaction
SHOW GETS
mtopic = ALIAS()
SET TOPIC TO &mtopic
*:  EOF: KBEDIT.ac1
```

```
*
*
*       08/09/94            OBLIST.SPR              09:40:10
*
*
*       Author's Name
*
*       Copyright (c) 1994 Company Name
*       Address
*       City,     Zip
*
*       Description:
*       This program was automatically generated by GENSCRN.
*
*

PARAMETERS marray, m.item

*
*
*            OBLIST/MS-DOS Setup Code - SECTION 1
*
*
*

#REGION 1
EXTERNAL ARRAY marray


#REGION 0
REGIONAL m.currarea, m.talkstat, m.compstat

IF SET("TALK") = "ON"
   SET TALK OFF
   m.talkstat = "ON"
ELSE
   m.talkstat = "OFF"
ENDIF
m.compstat = SET("COMPATIBLE")
SET COMPATIBLE FOXPLUS

*
*
*                 MS-DOS Window definitions
*
*
*

IF NOT WEXIST("w_obj") ;
      OR UPPER(WTITLE("W_OBJ")) == "W_OBJ.PJX" ;
      OR UPPER(WTITLE("W_OBJ")) == "W_OBJ.SCX" ;
      OR UPPER(WTITLE("W_OBJ")) == "W_OBJ.MNX" ;
      OR UPPER(WTITLE("W_OBJ")) == "W_OBJ.PRG" ;
      OR UPPER(WTITLE("W_OBJ")) == "W_OBJ.FRX" ;
      OR UPPER(WTITLE("W_OBJ")) == "W_OBJ.QPR"
   DEFINE WINDOW w_obj ;
      FROM INT((SROW()-21)/2),INT((SCOL()-62)/2) ;
      TO INT((SROW()-21)/2)+20,INT((SCOL()-62)/2)+61 ;
      TITLE "Object list" ;
      NOFLOAT ;
      NOCLOSE ;
      SHADOW ;
      NOMINIMIZE ;
      COLOR SCHEME 1
ENDIF


*
*
*              OBLIST/MS-DOS Screen Layout
*
*
*

#REGION 1
IF WVISIBLE("w_obj")
   ACTIVATE WINDOW w_obj SAME
ELSE
   ACTIVATE WINDOW w_obj NOSHOW
ENDIF
@ 0,1 GET m.obj ;
   PICTURE "@&N" ;
   FROM marray ;
   SIZE 17,58 ;
   DEFAULT 1 ;
   COLOR SCHEME 2
@ 18,19 GET m.buttons ;
   PICTURE "@*HT \<New;\<Ok;\<Cancel" ;
   SIZE 1,8,1 ;
   DEFAULT 1 ;
   VALID _qsf0kq4m3()

IF NOT WVISIBLE("w_obj")
   ACTIVATE WINDOW w_obj
ENDIF

READ CYCLE

RELEASE WINDOW w_obj

#REGION 0
IF m.talkstat = "ON"
   SET TALK ON
ENDIF
IF m.compstat = "ON"
   SET COMPATIBLE ON
ENDIF


*
*
*              OBLIST/MS-DOS Cleanup Code
*
*
*

#REGION 1
*FUNCTION validf
*IF !seek(id,'quals')
*   RETURN right(str(id),5) + "  " + name
*ENDIF
*RETURN

*
*
*       _QSF0KQ4M3           m.buttons VALID
*
*       Function Origin:
*
*       From Platform:       MS-DOS
*       From Screen:         OBLIST,     Record Number:    3
*       Variable:          m.buttons
*       Called By:         VALID Clause
*       Object Type:       Push Button
*       Snippet Number:    1
*
*
*
FUNCTION _qsf0kq4m3     &&  m.buttons VALID
#REGION 1
DO CASE
CASE m.buttons = 1
   m.item = ""
CASE m.buttons = 2
   m.item = marray[m.obj]
OTHERWISE
   m.item ="~"
ENDCASE
*:  EOF: OBLIST.ac1
```

```
*
*
*       08/09/94              DDEDIT.SPR              09:40:13
*
*
*       Author's Name
*
*       Copyright (c) 1994 Company Name
*       Address
*       City,     Zip
*
*       Description:
*       This program was automatically generated by GENSCRN.
*
*

PARAMETERS m.newid, m.newnm

*
*
*               DDEDIT/Windows Setup Code - SECTION 1
*
*
*

#REGION 1
PRIVATE mp, mpn, msel, m.adding
DIMENSION enum[20], enumr[20], mrules[1]
msel = SELECT()
SELECT dict
SET ORDER TO 1
enum = " "
enumr = 0
mrules = " "
mp = 1
m.adding = .F.
m.qrecno = RECNO()
IF PARAMETER() = 0
   m.newid = id
ENDIF

SEEK m.newid
IF !FOUND()
   SCATTER MEMVAR BLANK
   m.adding = .T.
   m.id = m.newid
   m.name = m.newnm
ELSE
   SCATTER MEMVAR
ENDIF
IF !m.adding
   DO setrules
ENDIF
DO setenum


#REGION 0
REGIONAL m.currarea, m.talkstat, m.compstat

IF SET("TALK") = "ON"
   SET TALK OFF
   m.talkstat = "ON"
ELSE
   m.talkstat = "OFF"
ENDIF
m.compstat = SET("COMPATIBLE")
SET COMPATIBLE FOXPLUS

m.rborder = SET("READBORDER")
SET readborder ON

*
*
*                    Windows Window definitions
*
*
*

IF NOT WEXIST("_qsf0kq5ys")
   DEFINE WINDOW _qsf0kq5ys ;
      AT  0.000, 0.000  ;
      SIZE 33.083,71.750 ;
      TITLE "Data Element Editor" ;
      FONT "Terminal", 8 ;
      FLOAT ;
      NOCLOSE ;
      MINIMIZE ;
      SYSTEM ;
      COLOR RGB(,,,128,128,128)
   MOVE WINDOW _qsf0kq5ys CENTER
ENDIF


*
*
*                 DDEDIT/Windows Screen Layout
*
*
*

#REGION 1
IF WVISIBLE("_qsf0kq5ys")
   ACTIVATE WINDOW _qsf0kq5ys SAME
ELSE
   ACTIVATE WINDOW _qsf0kq5ys NOSHOW
ENDIF
@ 1.333,57.750 SAY "Type"  ;
   FONT "Terminal", 8
@ 29.167,12.000 SAY "Hi"  ;
   FONT "Terminal", 8
@ 29.167,31.500 SAY "Lo"  ;
   FONT "Terminal", 8
@ 29.333,49.500 SAY "Units"  ;
   FONT "Terminal", 8
@ 1.333,4.500 SAY "Id"  ;
   FONT "Terminal", 8
@ 1.417,13.500 SAY "Name"  ;
   FONT "Terminal", 8
@ 27.167,9.000 SAY "Width"  ;
   FONT "Terminal", 8
@ 27.167,37.500 SAY "Decimals"  ;
   FONT "Terminal", 8
@ 29.083,4.500 SAY "Range:"  ;
   FONT "Terminal", 8
@ 17.167,4.500 SAY "Enum"  ;
   FONT "Terminal", 8
@ 27.167,4.500 SAY "Val:"  ;
   FONT "Terminal", 8
@ 11.333,4.500 SAY "Question Askable:"  ;
   FONT "Terminal", 8 ;
   STYLE "T"
```

```
@ 3.167,4.500 SAY "Use in rules:"  ;
   FONT "Terminal", 8 ;
   STYLE "T"
@ 1.167,7.750 GET m.id ;
   SIZE 1.000,3.125 ;
   DEFAULT " " ;
   FONT "Terminal", 8 ;
   DISABLE
@ 1.167,19.750 GET m.name ;
   SIZE 1.000,33.875 ;
   DEFAULT " " ;
   FONT "Terminal", 8
@ 1.083,63.000 GET m.datatype ;
   PICTURE "@^ N;E;M;L" ;
   SIZE 1.500,4.375 ;
   DEFAULT "N" ;
   FONT "Terminal", 8 ;
   VALID _qsf0kq6m1() ;
   DISABLE
@ 5.167,4.625 GET m.rulesuse ;
   PICTURE "@&N" ;
   FROM mrules ;
   SIZE 4.667,62.875 ;
   DEFAULT 1 ;
   FONT "Terminal", 8
@ 11.333,22.750 GET m.askable ;
   SIZE 1.000,4.625 ;
   DEFAULT .F. ;
   FONT "Terminal", 8
@ 13.167,4.750 EDIT m.question ;
   SIZE 3.000,62.875,0.000 ;
   DEFAULT " " ;
   FONT "Terminal", 8 ;
   SCROLL ;
   WHEN m.askable
@ 19.167,4.625 GET mp ;
   PICTURE "@&N" ;
   FROM enum ;
   SIZE 7.000,62.875 ;
   DEFAULT 1 ;
   FONT "Terminal", 8 ;
   WHEN m.datatype $ "EML" ;
   VALID _qsf0kq6t0()
@ 27.167,16.750 GET m.width ;
   SIZE 1.000,13.000 ;
   DEFAULT 0 ;
   FONT "Terminal", 8 ;
   WHEN m.datatype = "N" ;
   DISABLE
@ 27.167,49.750 GET m.dec ;
   SIZE 1.000,16.000 ;
   DEFAULT 0 ;
   FONT "Terminal", 8 ;
   WHEN m.datatype = "N" ;
   DISABLE
@ 29.167,16.750 GET m.hi ;
   SIZE 1.000,13.000 ;
   DEFAULT 0 ;
   FONT "Terminal", 8 ;
   WHEN m.datatype = "N" ;
   DISABLE
@ 29.167,34.750 GET m.lo ;
   SIZE 1.000,13.000 ;
   DEFAULT 0 ;
   FONT "Terminal", 8 ;
   WHEN m.datatype = "N" ;
   DISABLE
@ 29.333,55.750 GET m.units ;
   SIZE 1.000,10.000 ;
   DEFAULT " " ;
   FONT "Terminal", 8 ;
   WHEN m.datatype = "N" ;
   DISABLE
@ 31.000,27.000 GET mbuttons ;
   PICTURE "@*HT OK;Cancel" ;
   SIZE 1.917,8.375,0.750 ;
   DEFAULT 1 ;
   FONT "Terminal", 8 ;
   VALID _qsf0kq6yy()

┌IF NOT WVISIBLE("_qsf0kq5ys")
│   ACTIVATE WINDOW _qsf0kq5ys
└ENDIF

READ CYCLE ;
   WHEN _qsf0kq74f()

RELEASE WINDOW _qsf0kq5ys

#REGION 0

SET readborder &rborder

┌IF m.talkstat = "ON"
│   SET TALK ON
└ENDIF
┌IF m.compstat = "ON"
│   SET COMPATIBLE ON
└ENDIF


*
*
*                 DDEDIT/Windows Cleanup Code
*
*
*

#REGION 1
SELECT dict
<——RETURN

*----------------------------------------------------------------
*
* This procedure set the list of rules occupied the object
*
*----------------------------------------------------------------


*
*
*        DDEDIT/Windows Supporting Procedures and Functions
*
*
*

#REGION 1
FUNCTION setrules
PRIVATE msel,mvalid
msel = SELECT()
SELECT quals
SET FILTER TO id = m.id AND OBJECT = "S"
```

```
COUNT TO mcount
IF mcount > 0
   DIMENSION mrules[mcount]
   i = 0
   SCAN
      i = i + 1
      mrules[i] = areaname(area) + "-- rules: " + ALLTRIM(STRTRAN(rul
=> es,"|",","))
      IF !EMPTY(ruleso)
         mrules[i] = mrules[i] + IIF(EMPTY(rules),"",",") ;
            + ALLTRIM(STRTRAN(ruleso,"|",","))
      ENDIF
   ENDSCAN
ELSE
   DIMENSION mrules[1]
   mrules[1]=""
ENDIF
SET FILTER TO
SELECT (msel)
RETURN

PROCEDURE edenum
SELECT enum
IF enumr[mp] > 0
   GOTO enumr[mp]
ENDIF
*m.id = dict.id
DO ddenum.spr
DO setenum
SHOW GETS
SELECT dict
RETURN


PROCEDURE setenum
enum = " "
enumr = 0
IF m.datatype $ "EML"
   PRIVATE msel, i
   msel = SELECT()
   SELECT enum
   SEEK m.id
   i = 0
   DO WHILE id = m.id
      i = i + 1
      enumr[i] = RECNO()
      enum[i] = mutex + " " + TRIM( enumerate )
      SKIP
   ENDDO
   mpn = i
   mp = MIN(mpn, mp)
   mp = MAX(1, mp)
   IF enumr[mp] > 0
      GOTO enumr[mp]
   ENDIF
   SELECT (msel)
ENDIF
RETURN

PROCEDURE setval
PRIVATE m.dictid
IF m.datatype $ "N"
   PRIVATE msel, i
   msel = SELECT()
   SELECT VAL
   m.dictid = m.id
   SEEK m.id
   IF FOUND()
      SCATTER MEMVAR
   ELSE
      SCATTER MEMVAR BLANK
      m.id = m.dictid
   ENDIF
   SHOW GETS
   SELECT (msel)
ENDIF
RETURN

FUNCTION cleanenum
PRIVATE msel
msel = SELECT()
SELECT enum
SEEK m.id
IF FOUND()
   DELETE FOR id = m.id
   PACK
ENDIF
SELECT (msel)
RETURN

FUNCTION areaname
PARAMETER mid
PRIVATE msel,mname

msel = SELECT()
SELECT area
SEEK mid
IF FOUND()
   mname = name
ELSE
   mname = ""
ENDIF
SELECT (msel)
RETURN mname



*
*
*      _QSFOKQ6M1             m.datatype VALID
*
*      Function Origin:
*
*      From Platform:         Windows
*      From Screen:           DDEDIT,     Record Number:   17
*      Variable:              m.datatype
*      Called By:             VALID Clause
*      Object Type:           Popup
*      Snippet Number:        1
*
*
*
FUNCTION _qsf0kq6m1     &&  m.datatype VALID
#REGION 1
DO CASE
CASE m.datatype $ "N"
   DO setval
   enum = " "
   SHOW GET mp DISABLE
   SHOW GET m.width ENABLE
   SHOW GET m.dec ENABLE
   SHOW GET m.hi ENABLE
```

```
       SHOW GET m.lo ENABLE
       SHOW GET m.units ENABLE
       SHOW GETS
    OTHERWISE
       DO setenum
     IF EMPTY(enum[1])
          DO ddenum.spr WITH .T.
          DO setenum
     ENDIF
       SHOW GET mp ENABLE
       SHOW GET m.width DISABLE
       SHOW GET m.dec DISABLE
       SHOW GET m.hi DISABLE
       SHOW GET m.lo DISABLE
       SHOW GET m.units DISABLE
       SHOW GETS
    ENDCASE

 *
 *
 *         _QSF0KQ6T0          mp VALID
 *
 *         Function Origin:
 *
 *         From Platform:      Windows
 *         From Screen:        DDEDIT,     Record Number:   21
 *         Variable:           mp
 *         Called By:          VALID Clause
 *         Object Type:        List
 *         Snippet Number:     2
 *
 *
 *
 FUNCTION _qsf0kq6t0     &&  mp VALID
 #REGION 1
 DO edenum

 *
 *
 *         _QSF0KQ6YY          mbuttons VALID
 *
 *         Function Origin:
 *
 *         From Platform:      Windows
 *         From Screen:        DDEDIT,     Record Number:   27
 *         Variable:           mbuttons
 *         Called By:          VALID Clause
 *         Object Type:        Push Button
 *         Snippet Number:     3
 *
 *
 *
 FUNCTION _qsf0kq6yy     &&  mbuttons VALID
 #REGION 1
 DO CASE
 CASE mbuttons = 1    && ok
    SELECT dict
  IF m.adding
       APPEND BLANK
  ENDIF
    GATHER MEMVAR
  IF m.datatype $ "N"
     = cleanenum()
       SELECT VAL
     IF m.adding
          APPEND BLANK
     ENDIF
       GATHER MEMVAR
  ENDIF
 CASE mbuttons = 2    && cancel
  IF m.adding
     = cleanenum()
  ENDIF
    mok = .F.
 ENDCASE


 *
 *
 *         _QSF0KQ74F          Read Level When
 *
 *         Function Origin:
 *
 *
 *         From Platform:      Windows
 *         From Screen:        DDEDIT
 *         Called By:          READ Statement
 *         Snippet Number:     4
 *
 *
 *
 FUNCTION _qsf0kq74f     && Read Level When
 *
 * When Code from screen: DDEDIT
 *
 #REGION 1
 IF m.adding
    SHOW GET m.datatype ENABLE
    SHOW GET m.rulesuse DISABLE
 ENDIF
 IF m.datatype $ "N"
    PRIVATE msel
    msel = SELECT()
    SELECT VAL
    SCATTER MEMVAR
    SHOW GET mp DISABLE
    SHOW GET m.width ENABLE
    SHOW GET m.dec ENABLE
    SHOW GET m.hi ENABLE
    SHOW GET m.lo ENABLE
    SHOW GET m.units ENABLE
    SHOW GETS
 ENDIF
*:  EOF: DDEDIT.ac1
```

```
*
*
*   08/09/94            DDENUM.SPR              09:40:17
*
*
*   Author's Name
*
*   Copyright (c) 1994 Company Name
*   Address
*   City,     Zip
*
*   Description:
*   This program was automatically generated by GENSCRN.
*
*

PARAMETERS m.new

*
*
*          DDENUM/Windows Setup Code - SECTION 1
*
*
*

#REGION 1
m.dictid = m.id
IF PARAMETER() = 0
   m.enumadd = .F.
   SCATTER MEMVAR
ELSE
   m.dictid = m.id
   m.enumadd = .T.
   SELECT enum
   SEEK m.dictid
   IF !FOUND()
      m.enumadd = .T.
      m.ord = 1
      m.id = m.dictid
      m.mutex = "-"
   ELSE
      SCATTER MEMVAR
   ENDIF
ENDIF
SHOW GETS

#REGION 0
REGIONAL m.currarea, m.talkstat, m.compstat

IF SET("TALK") = "ON"
   SET TALK OFF
   m.talkstat = "ON"
ELSE
   m.talkstat = "OFF"
ENDIF
m.compstat = SET("COMPATIBLE")
SET COMPATIBLE FOXPLUS

m.rborder = SET("READBORDER")
SET readborder ON

*
*
*          Windows Window definitions
*
*
*

IF NOT WEXIST("_qsf0kq96z")
   DEFINE WINDOW _qsf0kq96z ;
      AT 20.750, 7.500 ;
      SIZE 9.077,98.200 ;
      FONT "MS Sans Serif", 8 ;
      FLOAT ;
      NOCLOSE ;
      MINIMIZE ;
      SYSTEM
ENDIF


*
*
*          DDENUM/Windows Screen Layout
*
*
*

#REGION 1
IF WVISIBLE("_qsf0kq96z")
   ACTIVATE WINDOW _qsf0kq96z SAME
ELSE
   ACTIVATE WINDOW _qsf0kq96z NOSHOW
ENDIF
@ 5.154,5.000 SAY "Ord"  ;
   FONT "Terminal", 8
@ 1.462,4.800 SAY "Mutex"  ;
   FONT "Terminal", 8
@ 3.308,4.800 SAY "Enum"  ;
   FONT "Terminal", 8
@ 5.154,14.000 GET m.ord ;
   SIZE 1.000,2.000 ;
   DEFAULT " " ;
   FONT "Terminal", 8 ;
   DISABLE
@ 1.615,14.400 GET m.mutex ;
   SIZE 1.000,1.000 ;
   DEFAULT " " ;
   FONT "Terminal", 8 ;
   VALID _qsf0kq9mb()
@ 3.231,14.400 GET m.enumerate ;
   SIZE 1.000,47.750 ;
   DEFAULT " " ;
   FONT "Terminal", 8 ;
   PICTURE "@!"
@ 6.462,26.400 GET mbuttons ;
   PICTURE "@*HN Add;OK;Cancel" ;
   SIZE 1.917,7.500,0.750 ;
   DEFAULT 1 ;
   FONT "Terminal", 8 ;
   VALID _qsf0kq9qc()

IF NOT WVISIBLE("_qsf0kq96z")
   ACTIVATE WINDOW _qsf0kq96z
ENDIF

READ CYCLE

RELEASE WINDOW _qsf0kq96z

#REGION 0

```

```
    SET readborder &rborder

  IF m.talkstat = "ON"
     SET TALK ON
  ENDIF
  IF m.compstat = "ON"
     SET COMPATIBLE ON
  ENDIF

    *
    *
    *      _QSFOKQ9MB           m.mutex VALID
    *
    *      Function Origin:
    *
    *      From Platform:       Windows
    *      From Screen:         DDENUM,     Record Number:    6
    *      Variable:            m.mutex
    *      Called By:           VALID Clause
    *      Object Type:         Field
    *      Snippet Number:      1
    *
    *
    *
    FUNCTION _qsf0kq9mb     &&  m.mutex VALID
    #REGION 1
   DO CASE
   CASE m.datatype $ "EL"
<----- RETURN m.mutex = "-"
   CASE m.datatype $ "M"
<----- RETURN m.mutex $ "-+"
   ENDCASE
<--RETURN .F.

    *
    *
    *      _QSFOKQ9QC           mbuttons VALID
    *
    *      Function Origin:
    *
    *      From Platform:       Windows
    *      From Screen:         DDENUM,     Record Number:    8
    *      Variable:            mbuttons
    *      Called By:           VALID Clause
    *      Object Type:         Push Button
    *      Snippet Number:      2
    *
    *
    *
    FUNCTION _qsf0kq9qc     &&  mbuttons VALID
    #REGION 1
   DO CASE
   CASE mbuttons = 1   && add
      IF m.enumadd
         APPEND BLANK
         GATHER MEMVAR
      ENDIF
      m.enumadd = .T.
      m.neword = 0
      DO WHILE id = m.dictid
         m.neword = ord
         SKIP
      ENDDO
      SCATTER MEMVAR BLANK
      m.ord  =  m.neword + 1
      m.id  =  m.dictid
      m.mutex  =  "-"
       _CUROBJ = OBJNUM(m.mutex)
      SHOW GETS
   CASE mbuttons = 2    && ok
      IF m.enumadd
         APPEND BLANK
      ENDIF
      mok = .T.
      GATHER MEMVAR
      CLEAR READ
   CASE mbuttons = 3    && cancel
      mok = .F.
      CLEAR READ
   ENDCASE
*:  EOF: DDENUM.ac1
```

```
*
*
*    08/09/94             KBLOAD.SPR             09:40:19
*
*
*    Author's Name
*
*    Copyright (c) 1994 Company Name
*    Address
*    City,     Zip
*
*    Description:
*    This program was automatically generated by GENSCRN.
*
*

DO CASE
CASE _WINDOWS


   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
   ENDIF
   m.compstat = SET("COMPATIBLE")
   SET COMPATIBLE FOXPLUS

   m.rborder = SET("READBORDER")
   SET readborder ON

   *
   *
   *                Windows Window definitions
   *
   *
   *

   IF NOT WEXIST("_qsf0kqb51")
      DEFINE WINDOW _qsf0kqb51 ;
         AT  0.000, 0.000  ;
         SIZE 22.333,45.500 ;
         TITLE "Knowledge Base Loader" ;
         FONT "Terminal", 8 ;
         FLOAT ;
         NOCLOSE ;
         SHADOW ;
         NOMINIMIZE ;
         DOUBLE ;
         COLOR RGB(,,,128,128,128)
      MOVE WINDOW _qsf0kqb51 CENTER
   ENDIF


   *
   *
   *          KBLOAD/Windows Setup Code - SECTION 2
   *
   *
   *

   #REGION 1
   m.home = CURDIR()
   m.drv   = m.new
   m.src   = ""
   m.fil   = ""
   m.ok = .F.
   SET DEFA TO (m.data)
   DEFINE POPUP listdir ;
      PROMPT FILES LIKE *. ;
      SCROLL ;
      MARGIN ;
      MARK " "
   ON ERROR m.src =""

   *
   *
   *          KBLOAD/Windows Screen Layout
   *
   *
   *

   #REGION 1
   IF WVISIBLE("_qsf0kqb51")
      ACTIVATE WINDOW _qsf0kqb51 SAME
   ELSE
      ACTIVATE WINDOW _qsf0kqb51 NOSHOW
   ENDIF
   @ 2.500,0.875 GET m.src ;
      SIZE 1.333,32.125 ;
      DEFAULT " " ;
      FONT "Terminal", 8 ;
      VALID _qsf0kqbnk()
   @ 7.583,1.375 GET m.new ;
      SIZE 1.167,31.750 ;
      DEFAULT " " ;
      FONT "Terminal", 8 ;
      VALID _qsf0kqbt0() ;
      DISABLE
   @ 9.917,1.000 GET m.dbf ;
      PICTURE "@&N" ;
      POPUP listdir ;
      SIZE 11.667,32.625 ;
      DEFAULT " " ;
      FONT "Terminal", 8 ;
      STYLE "" ;
      WHEN _qsf0kqc1j() ;
      VALID _qsf0kqc42()
   @ 14.250,34.875 GET m.load ;
      PICTURE "@*VN Load" ;
      SIZE 2.083,7.750,1.083 ;
      DEFAULT 1 ;
      FONT "Terminal", 8 ;
      WHEN loadok() ;
      VALID _qsf0kqc6t() ;
      DISABLE
   @ 17.417,34.875 GET mbuttonc ;
      PICTURE "@*VT Cancel" ;
      SIZE 2.250,7.875,1.083 ;
      DEFAULT 1 ;
      FONT "Terminal", 8
   @ 1.250,0.750 SAY "Read From Definition file:"  ;
      FONT "Terminal", 8
   @ 6.083,0.750 SAY "Create Files in Temporary directory:"  ;
      FONT "Terminal", 8

   IF NOT WVISIBLE("_qsf0kqb51")
      ACTIVATE WINDOW _qsf0kqb51
   ENDIF

   READ CYCLE MODAL

   RELEASE WINDOW _qsf0kqb51

   #REGION 0

   SET readborder &rborder

   IF m.talkstat = "ON"
      SET TALK ON
   ENDIF
   IF m.compstat = "ON"
      SET COMPATIBLE ON
   ENDIF


   *
   *
   *          KBLOAD/Windows Cleanup Code
   *
   *
   *

   #REGION 1
   SET DEFA TO (m.home)
   ON ERROR
   * load enable

CASE _DOS


   #REGION 0
   REGIONAL m.currarea, m.talkstat, m.compstat

   IF SET("TALK") = "ON"
      SET TALK OFF
      m.talkstat = "ON"
   ELSE
      m.talkstat = "OFF"
   ENDIF
   m.compstat = SET("COMPATIBLE")
   SET COMPATIBLE FOXPLUS


   *
   *
   *          MS-DOS Window definitions
   *
   *
   *

IF NOT WEXIST("_qsf0kqci7")
   DEFINE WINDOW _qsf0kqci7 ;
      FROM INT((SROW()-18)/2),INT((SCOL()-55)/2) ;
      TO INT((SROW()-18)/2)+17,INT((SCOL()-55)/2)+54 ;
      TITLE "Knowledge Base Loader" ;
      FLOAT ;
      NOCLOSE ;
      SHADOW ;
      NOMINIMIZE ;
      DOUBLE ;
      COLOR SCHEME 5
ENDIF


   *
   *
   *          KBLOAD/MS-DOS Setup Code - SECTION 2
   *
   *
   *

   #REGION 1
   m.home = CURDIR()
   m.drv   = m.new
   m.src   = ""
   m.fil   = ""
   m.ok = .F.
   SET DEFA TO (m.data)
   DEFINE POPUP listdir ;
      PROMPT FILES LIKE *. ;
      SCROLL ;
      MARGIN ;
      MARK " "
   ON ERROR m.src =""

   *
   *
   *          KBLOAD/MS-DOS Screen Layout
   *
   *
   *

   #REGION 1
```

```
IF WVISIBLE("_qsf0kqci7")
   ACTIVATE WINDOW _qsf0kqci7 SAME
ELSE
   ACTIVATE WINDOW _qsf0kqci7 NOSHOW
ENDIF
 @ 2,1 GET m.src ;
   SIZE 1,30 ;
   DEFAULT " " ;
   VALID _qsf0kqcp2()
 @ 6,1 GET m.new ;
   SIZE 1,30 ;
   DEFAULT " " ;
   VALID _qsf0kqcro() ;
   DISABLE
 @ 7,1 GET m.dbf ;
   PICTURE "@&N" ;
   POPUP listdir ;
   SIZE 9,30 ;
   DEFAULT " " ;
   WHEN _qsf0kqcua() ;
   VALID _qsf0kqcwt() ;
   COLOR SCHEME 6
 @ 13,42 GET m.load ;
   PICTURE "@*VT Load" ;
   SIZE 1,8,1 ;
   DEFAULT 1 ;
   WHEN loadok() ;
   VALID _qsf0kqczg() ;
   DISABLE
 @ 15,42 GET mbuttonc ;
   PICTURE "@*VT Cancel" ;
   SIZE 1,8,1 ;
   DEFAULT 1
 @ 1,1 SAY "Read From Definition file:" ;
   SIZE 1,26, 0
 @ 5,1 SAY "Create Files in Temporary directory:" ;
   SIZE 1,36, 0

IF NOT WVISIBLE("_qsf0kqci7")
   ACTIVATE WINDOW _qsf0kqci7
ENDIF

 READ CYCLE MODAL

 RELEASE WINDOW _qsf0kqci7

 #REGION 0
IF m.talkstat = "ON"
   SET TALK ON
ENDIF
IF m.compstat = "ON"
   SET COMPATIBLE ON
ENDIF


  *
  *
  *          KBLOAD/MS-DOS Cleanup Code
  *
  *
  *

     #REGION 1
     SET DEFA TO (m.home)
     ON ERROR
     * load enable


ENDCASE


*
*
*       _QSFOKQBNK           m.src VALID
*
*       Function Origin:
*
*       From Platform:       Windows
*       From Screen:         KBLOAD,     Record Number:    2
*       Variable:            m.src
*       Called By:           VALID Clause
*       Object Type:         Field
*       Snippet Number:      1
*
*
*
FUNCTION _qsf0kqbnk     &&  m.src VALID
#REGION 1
m.src = LOCFILE(m.src,"","Locate Definition File")
=loadok()


*
*
*       _QSFOKQBTO           m.new VALID
*
*       Function Origin:
*
*       From Platform:       Windows
*       From Screen:         KBLOAD,     Record Number:    3
*       Variable:            m.new
*       Called By:           VALID Clause
*       Object Type:         Field
*       Snippet Number:      2
*
*
*
FUNCTION _qsf0kqbt0     &&  m.new VALID
#REGION 1
=loadok()

*
*
*       _QSFOKQC1J           m.dbf WHEN
*
*       Function Origin:
*
*       From Platform:       Windows
*       From Screen:         KBLOAD,     Record Number:    4
*       Variable:            m.dbf
*       Called By:           WHEN Clause
*       Object Type:         List
*       Snippet Number:      3
*
*
*
FUNCTION _qsf0kqc1j     &&  m.dbf WHEN
```

```
#REGION 1
m.new = PRMBAR("listdir",2)
┌IF FILE(m.new + "\nul")
│   SHOW OBJECT OBJNUM(m.new) ENABLE
├ELSE
│   SHOW OBJECT OBJNUM(m.new) DISABLE
└ENDIF
=loadok()

*
*
*      _QSFOKQC42            m.dbf VALID
*
*      Function Origin:
*
*      From Platform:        Windows
*      From Screen:          KBLOAD,      Record Number:    4
*      Variable:             m.dbf
*      Called By:            VALID Clause
*      Object Type:          List
*      Snippet Number:       4
*
*
*
FUNCTION _qsf0kqc42     &&  m.dbf VALID
#REGION 1
=loadok()

*
*
*      _QSFOKQC6T            m.load VALID
*
*      Function Origin:
*
*      From Platform:        Windows
*      From Screen:          KBLOAD,      Record Number:    5
*      Variable:             m.load
*      Called By:            VALID Clause
*      Object Type:          Push Button
*      Snippet Number:       5
*
*
*
FUNCTION _qsf0kqc6t     &&  m.load VALID
#REGION 1
m.srcbak = m.src
m.valid = checkfile(m.new)
┌IF !m.valid
│   SHOW GET m.load DISABLE
│   _CUROBJ = OBJNUM(m.dbf)
│   m.src = m.srcbak
│   SHOW GETS
<───RETURN
└ENDIF
olderr = ON("error")
ON ERROR RETURN
mrun = m.src + " " + m.new
m.temp = CURDIR()
m.t1 = LOCFILE("rulex","exe","where is file RULEX.EXE")
m.t1 = SUBSTR(m.t1, 1, AT("RULEX.EXE",m.t1) -1)
SET DEFA TO (m.t1)
RUN /400 rulex &mrun
SET DEFA TO (m.temp)
= popupshow("Working....")
DO kbldr WITH m.new
= popuphide()
ON ERROR &olderr
CLEAR READ


*
*
*      _QSFOKQCP2            m.src VALID
*
*      Function Origin:
*
*      From Platform:        MS-DOS
*      From Screen:          KBLOAD,      Record Number:   11
*      Variable:             m.src
*      Called By:            VALID Clause
*      Object Type:          Field
*      Snippet Number:       6
*
*
*
FUNCTION _qsf0kqcp2     &&  m.src VALID
#REGION 1
m.src = LOCFILE(m.src)
=loadok()


*
*
*      _QSFOKQCRO            m.new VALID
*
*      Function Origin:
*
*      From Platform:        MS-DOS
*      From Screen:          KBLOAD,      Record Number:   12
*      Variable:             m.new
*      Called By:            VALID Clause
*      Object Type:          Field
*      Snippet Number:       7
*
*
*
FUNCTION _qsf0kqcro     &&  m.new VALID
#REGION 1
=loadok()

*
*
*      _QSFOKQCUA            m.dbf WHEN
*
*      Function Origin:
*
*      From Platform:        MS-DOS
*      From Screen:          KBLOAD,      Record Number:   13
*      Variable:             m.dbf
*      Called By:            WHEN Clause
*      Object Type:          List
*      Snippet Number:       8
*
*
*
FUNCTION _qsf0kqcua     &&  m.dbf WHEN
#REGION 1
m.new = PRMBAR("listdir",2)
┌IF FILE(m.new + "\nul")
│   SHOW OBJECT OBJNUM(m.new) ENABLE
├ELSE
│   SHOW OBJECT OBJNUM(m.new) DISABLE
```

```
└ENDIF
 =loadok()

 *
 *      _QSF0KQCWT          m.dbf VALID
 *
 *      Function Origin:
 *
 *      From Platform:      MS-DOS
 *      From Screen:        KBLOAD,     Record Number:   13
 *      Variable:           m.dbf
 *      Called By:          VALID Clause
 *      Object Type:        List
 *      Snippet Number:     9
 *
 *
 *
 FUNCTION _qsf0kqcwt     &&  m.dbf VALID
 #REGION 1
 =loadok()

 *
 *
 *      _QSF0KQCZG          m.load VALID
 *
 *      Function Origin:
 *
 *      From Platform:      MS-DOS
 *      From Screen:        KBLOAD,     Record Number:   14
 *      Variable:           m.load
 *      Called By:          VALID Clause
 *      Object Type:        Push Button
 *      Snippet Number:     10
 *
 *
 *
 FUNCTION _qsf0kqczg     &&  m.load VALID
 #REGION 1
 mrun = m.src + " " + m.new
 m.temp = CURDIR()
 m.t1 = LOCFILE("rulex.exe")
 m.t1 = SUBSTR(m.t1, 1, AT("RULEX.EXE",m.t1) -1)
 SET DEFA TO (m.t1)
 RUN /400 rulex &mrun
 SET DEFA TO (m.temp)
 DO kbldr WITH m.new



 *
 *
 *      KBLOAD/MS-DOS Supporting Procedures and Functions
 *
 *
 *

 #REGION 1

 *
 *
 *               KBLOAD Function LOADOK
 *
 *
 *


 PROCEDURE loadok
┌DO CASE
├CASE _DOS
│  m.ok = .T.
│ ┌IF EMPTY(m.src)
│ │  m.ok = .F.
│ └ENDIF
│  m.ok = m.ok .AND. FILE(m.src)
│  m.new = ALLTRIM(m.new)
│ ┌IF EMPTY(m.new)
│ │  m.ok = .F.
│ └ENDIF
│ ┌IF RIGHT(m.new, 1) != "\"
│ │  m.new = m.new + "\"
│ └ENDIF
│  m.d1 = IIF(RIGHT(m.data,1) != "\", m.data + "\", m.data)
│ ┌IF UPPER(FULLPATH(m.new)) == UPPER(FULLPATH(m.d1))
│ │  m.ok = .F.
│ └ENDIF
│ ┌IF !FILE(m.new + "nul")
│ │  m.ok = .F.
│ └ENDIF
│ ┌IF m.ok
│ │  SHOW OBJECT OBJNUM(m.load) ENABLE
│ ├ELSE
│ │  SHOW OBJECT OBJNUM(m.load) DISABLE
│ └ENDIF
<──────RETURN m.ok
├CASE _WINDOWS
│  m.ok = .T.
│ ┌IF EMPTY(m.src)
│ │  m.ok = .F.
│ └ENDIF
│  m.ok = m.ok .AND. FILE(m.src)
│  m.new = ALLTRIM(m.new)
│ ┌IF EMPTY(m.new)
│ │  m.ok = .F.
│ └ENDIF
│ ┌IF RIGHT(m.new, 1) != "\"
│ │  m.new = m.new + "\"
│ └ENDIF
│  m.d1 = IIF(RIGHT(m.data,1) != "\", m.data + "\", m.data)
│ ┌IF UPPER(FULLPATH(m.new)) == UPPER(FULLPATH(m.d1))
│ │  m.ok = .F.
│ └ENDIF
│ ┌IF !FILE(m.new + "nul")
│ │  m.ok = .F.
│ └ENDIF
│ ┌IF m.ok
│ │  SHOW OBJECT OBJNUM(m.load) ENABLE
│ ├ELSE
│ │  SHOW OBJECT OBJNUM(m.load) DISABLE
│ └ENDIF
<──────RETURN m.ok
│
└ENDCASE

  *
  *
  *               KBLOAD Function CHECKFILE
  *
  *
  *

  FUNCTION checkfile
```

```
    PARAMETER m.new
    PRIVATE m.exist, m.fload
    m.exist = .F.
  ┌─DO CASE
  ├─CASE FILE(m.new + "area.dbf")
  │    m.exist = .T.
  ├─CASE FILE(m.new + "rule.dbf")
  │    m.exist = .T.
  ├─CASE FILE(m.new + "premise.dbf")
  │    m.exist = .T.
  ├─CASE FILE(m.new + "fact.dbf")
  │    m.exist = .T.
  ├─CASE FILE(m.new + "action.dbf")
  │    m.exist = .T.
  ├─CASE FILE(m.new + "disease.dbf")
  │    m.exist = .T.
  ├─CASE FILE(m.new + "dict.dbf")
  │    m.exist = .T.
  ├─CASE FILE(m.new + "val.dbf")
  │    m.exist = .T.
  ├─CASE FILE(m.new + "enum.dbf")
  │    m.exist = .T.
  └─ENDCASE
    m.fload = .T.
  ┌─IF m.exist
  │    m.fload = yesno("Database in " + m.new + " already exist, overwrit
=> e & delete it?","YES","NO")
  └─ENDIF
<───RETURN m.fload

*:  EOF: KBLOAD.ac1
```

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE JUNE 1996	3. REPORT TYPE AND DATE COVERED FINAL JAN 1994-1995
4. TITLE AND SUBTITLE TECHNICAL MANUAL FOR THE NAVY COMPUTER ASSISTED MEDICAL DIAGNOSIS KNOWLEDGE BASE EDITOR (NCAMD-KBE), VERSION 1.0		5. FUNDING NUMBERS Program Element: 63706N Work Unit Number: M0095.005-6103
6. AUTHOR(S) HOA L. LY AND DIANNA M. PEARSALL		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Health Research Center P. O. Box 85122 San Diego, CA 92186-5122		8. PERFORMING ORGANIZATION TECHNICAL DOCUMENT 96-4D
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Naval Medical Research and Development Command National Naval Medical Center Building 1, Tower 2 Bethesda, MD 20889-5044		10. SPONSORING/MONITORING AGENCY REPORT NUMBER
11. SUPPLEMENTARY NOTES		
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.		12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This technical manual contains the information on the program source code, data elements, and the database structure needed to maintain the Navy Computer Assisted Medical Diagnosis Knowledge Base Editor (NCAMD-KBE). This documentation was created using the FOXDOC version 2.5a program.

14. SUBJECT TERMS Computer diagnosis Technical Manual			15. NUMBER OF PAGES 181 / 16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT Unlimited